LEARNING *with* PERSONAL COMPUTERS

Helga A. H. Rowe

LEARNING WITH PERSONAL COMPUTERS:

ISSUES, OBSERVATIONS AND PERSPECTIVES

Helga A.H. Rowe

with the assistance of
Irene Brown and Isabel Lesman

ACER

First published 1993
by The Australian Council for Educational Research Ltd
Radford House, Frederick Street, Hawthorn, Victoria 3122, Australia

Printed by Allanby Press Printers Pty Ltd

National Library of Australia
Cataloguing-in-Publication data:

Rowe, Helga A.H. (Helga Anneliese Hildegard).
Learning with personal computers.

Includes index.
ISBN 0 86431 129 X.

1. Microcomputers - Study and teaching. 2. Computer-assisted instruction. I. Australian Council for Educational Research. II. Title.

371.334

Contents

Preface

Australia is moving rapidly into the use of computers for learning and teaching in schools. Over a three year period (1984-86) the Commonwealth Government committed 18 million Australian dollars for the development and implementation of a National Computer Education Program. Since then further public and private funds have been raised for the purposes of educational computing by education departments, school systems and individual schools.

In 1984 the Commonwealth Schools Commission appointed a National Advisory Committee on Computers in Schools. The primary task of this committee was to provide advice to the Commonwealth Schools Commission on the development of a rationale and a plan for the use of computers in education. The principles developed to guide the introduction of computers into schools focused on two issues, namely that the programs be broadly based, and that they would lead to equality of outcome. A broadly based program was defined as one which focuses on teaching about and with computers. Equality of outcome was expected to be achieved by improving the access of all children to computers.

The Commonwealth Schools Commission expected the use of computers in schools to focus on a greater understanding by students of the effects which computers and information technology can have on society. It argued:

> An important function of schools is to help students to communicate, to think, to value and to act first and foremost as young people in school, and second in the context of the society in which they live. (Commonwealth Schools Commission, 1984, p. 21)

> The use of computers should be seen as an integral part of the teaching and learning process, and should not be regarded . . . as a potential replacement for teachers, nor should computer aided instruction become necessarily the focus of a school's use of computers. (Commonwealth Schools Commission, 1984, p. 22)

The Commission produced basic sets of aims and objectives for a National Computer Education Program for students, teachers, principals and administrators. The desired outcomes for students were specified as follows:

> Students should learn to
>
> use computers for inquiry, analysis, information processing, problem solving and recreation;
> make informed and responsible judgments about those aspects of computer use that affect them and others in economic, social, political and physical contexts;
> recognise the sort of problems that are not amenable to computer solutions;
> become conversant with the broad characteristics of the hardware and software with which they are working and acquire the capacity to make a consumer evaluation of such products; and
> undertake a formal study of information processing or appropriate aspects of it. (Commonwealth Schools Commission, 1984, p. 20)

In the years since, computer education has been taken up in all states, and some attempts have been made to meet and refine the Commission's aims and objectives, but the offerings of schools vary dramatically. Reasons for this include insufficient hardware, inability to obtain suitable software, limitations in teacher education and lack of firm curriculum guidelines. However, the most important obstacle to the successful implementation of computing into schools is probably the lack of knowledge of how to link this technological innovation with teaching and how to integrate it into the day-to-day curriculum. The computer *per se* cannot produce effective learning, but students can learn effectively with computers.

Convinced of the importance of computing in education for all students, the Queensland Education Department seeks to reduce the lack of knowledge by means of a considerable number of impressive initiatives in the area of educational technology. One of these initiatives was to provide each of 115 Year 6 and Year 7 students with a personal laptop computer for use at school and at home throughout the school year. Though not formally specified, the questions the SUNRISE project took on related to how computers might best be utilised in the classroom, and how their influence can be identified in various social and cognitive contexts. These are practical problems for educational research and development, because it is reasonable to expect that some children might learn better with computers, that most students would develop new thinking strategies which are adaptive to a largely computer-based learning environment, and that individual differences would influence learning with computers differently from learning in the traditional classroom. These are also profound theoretical questions, because they imply that the learning environment is an integral part of students' cognitive processes and educational and personal development.

The SUNRISE project conducted by the Queensland Education Department at Coombabah Primary School, Gold Coast, Queensland, was the subject of the empirical study of learning and teaching with computers that is described in this book.

As part of its support for the development of such innovative efforts, their dissemination and promotion, the Australian Council for Educational Research contributed funds to this project. We are grateful to the Queensland Department of Education for the provision of access to the SUNRISE classrooms at Coombabah and for the partial funding of the research which resulted in this book.

Summary

This book is about learning and teaching with personal computers. It is aimed at teachers, trainee teachers, those responsible for pre-service and in-service training of teachers, school administrators and parents. The contents of this book should be of particular interest to teachers and school administrators who are planning to introduce computers into classrooms or are teaching students with computers for the first time.

A wide range of practical and theoretical issues is addressed in this volume because of the difficulty teachers and educational administrators are experiencing in obtaining the type of information provided here. The extensive list of references will help those wishing to obtain deeper knowledge in a particular area. An attempt was made to include relevant materials from many sources. The empirical study conducted in the SUNRISE classrooms at Coombabah, Queensland, serves to illustrate the issues discussed and to raise further questions. Although the empirical findings reported here are based on a study in which each student had his/her own laptop computer, readers will find that the results of this study and the deliberations of the book as a whole are equally applicable to classrooms where two or three students share one computer.

The chapters of this book are arranged into four sections. Part I provides a theoretical framework for learning and teaching with computers. Part II deals with issues relating to the acquisition of computer literacy. Part III describes the empirical study conducted with 115 Year 6 and Year 7 students, and Part IV deals with issues relating to the professional development of teachers who teach students with computers and with the evaluation of computer software by teachers.

PART I: THEORETICAL FRAMEWORK (CHAPTERS 1 AND 2)

The promise and impact of personal computers in the pursuit of the goal of turning Australia into a *clever country* and important issues relating to the evaluation of the effectiveness of computing in schools are addressed in Chapter 1. Some implications of the computer revolution are related to computing in schools. Computing is discussed within a framework of *cognitive technologies.* Human/machine relationships, more specifically the idea of adapting the technology to one's own purposes versus adapting oneself to the technology, and social interaction between students are discussed. Other topics raised in this chapter include: computing as a cultural component of the learning environment; computer literacy as personal capital; and empowerment through metacognition. The final section of this chapter stresses the importance of teachers if learning with computers is to be effective.

Chapter 2 deals with educational and cognitive benefits of computing. It is generally accepted that computers can boost people's productivity and efficiency in performance. What is less known is that computing can provide users with a *box of reconstructible tools* which can change the characteristics of the problems and the learning tasks themselves and hence lead to a restructuring of the processes of problem solving and learning. A major portion of this chapter deals with the cognitive benefits of learning programming. The focus is on the interface between problem solving strategies and important components of programming instruction. An example is provided of a set of steps towards initial mastery of programming. The importance for teachers to make learning with computers cognitively demanding for their students is stressed, and examples of cognitive demanding activities are provided, also by contrasting the characteristics of programming experts with those of students in the classroom. Only through cognitively demanding learning activities can we expect to foster higher order thinking, problem solving and learning.

PART II: ACQUIRING COMPUTER LITERACY (CHAPTERS 3 TO 5)

What is being advocated in the three chapters of this section is not that students become programmers in a technical sense, but that they regard computers as a natural and integral part of their lives and that they view what they learn in computing as a fabric of experiences and knowledge which can be woven into many activities in and out of school. Alternative educational philosophies and perceptions of the human/computer relationship will determine the specific uses made of hardware and software in the classroom. The tools themselves are extremely versatile and can support many and quite contradictory educational philosophies and objectives.

Chapter 3 begins with a broad discussion of curriculum objectives and computing policy. How computing will be introduced into a school depends very much on the school's definition of *computer literacy*. This concept is discussed with respect to other literacies and various types of definitions are provided. The implications of comprehensive and narrow definitions of computer literacy are explained for various components of instructional computing. Suggestions are made and examples are provided of how one might integrate the computer into the existing curriculum. The chapter ends with a discussion of how the technology could best function as a mediator of cognitive development.

Chapter 4 looks at the acquisition of computer knowledge and skills more from the point of view of the teacher and the classroom culture he/she might create. One theme which runs through this chapter is that of *control.* Learning and teaching with computers provide the opportunity to encourage students to take more responsibility for their own work, motivation and learning. Handing

over, sharing and accepting control can be difficult for both teachers and students. Specific suggestions are made relating to the encouragement of independent and self-regulated learning by students, as they are constructing their own knowledge. The chapter contains a brief discussion of *Logo*, as this is the language used in the SUNRISE classrooms which were the subject of the empirical study reported in Part IV, and a frequently chosen vehicle for teaching with computers in Australian schools. Most of this chapter addresses quite concrete issues relating to classroom organisation and the interaction with students.

Chapter 5 introduces questions of assessment and evaluation of learning with computers. It deals with expected outcomes and ways of monitoring product and process. Examples of possible assessment procedures are discussed briefly and ways of evaluating the computing efforts of students working in groups is discussed. An initial suggestion is made for the evaluation of student performance on programming related tasks. An example of an *objectives wheel* is provided which can help integrate computing knowledge and skills, with curriculum units or subjects. At this stage, the area of assessment and evaluation in learning and teaching with computers is probably the most neglected of all the areas addressed in this book.

PART III: THE EMPIRICAL STUDY (CHAPTERS 6 TO 8)

The chapters in this section provide the bulk of the information resulting from the empirical study of 115 students with their own laptop computers, i.e. 56 Year 6 and 59 Year 7 students in the SUNRISE classrooms at Coombabah Primary School, Gold Coast, Queensland. Some results of this study are reported as examples or illustrations in relevant sections of the other chapters of this book.

Two major and severe limitations of the empirical study conducted at Coombabah must be recognised:

1 with increased availability of personal computers, improved software and related curriculum materials, and more professional development activities for teachers, the information presented here might become outdated quite quickly; and
2 a study based on observations made on 115 students from two classes in one non-randomly selected school certainly lacks reliability and thus generalisability.

Nevertheless, in view of the current large knowledge gap in the area of educational computing, information on learning and teaching with computers in Australia must be disseminated even if the findings are timebound. The experiences and reactions of even an unrepresentative sample of students and teachers are not only interesting and informative for teachers and educational

administrators who are planning to embark on similar projects, but also raise questions which must be addressed in research and practice. In the absence of similar research information collected in well designed and possibly comparative studies, the attitudes, knowledge, abilities and achievements, learning practices and preferences of students observed in two not too *untypical* classrooms are expected to provide some useful insights to the reader.

Chapter 6 describes the objectives, design and method of the empirical study. A preliminary conceptual framework, i.e. a system in which the variables under investigation might be related, for such a study is developed. Considerable attention is given to the methods and sources of data which are being collected. The identification of possible attributes of computer use for learning and their measurement are carefully considered. The attributes used in this study are defined and operationalised. An attempt was made to identify possible *indicators* of *effective* computer use by students. Though the data of the present study may not have warranted such careful analysis of student profiles of computer use, the reason for this part of the study was to attempt to come up with a way of looking at the attributes of learning in computing which might be used by the authors or other investigators in subsequent studies. In other words, this was an attempt to contribute to the development of methodology in a new area of educational research.

In a very tentative preliminary analysis of student profiles in computing processes, four groups of students were classified on the basis of their patterns of laptop use. The students in group A were labelled *orchestrators*. These students used the widest variety of learning applications closely linked with teacher and task demands, personal aims and skills, personal learning style and the social demands of the classroom. The attitudes of these students to computing and their uses of their computers reflect a harmony created by flexible and appropriate application. Students in group B were labelled *amplifiers*, because they tended to use the computer to amplify their existing skills, but viewed it as an adjunct, i.e. a non-essential but at times useful and convenient accessory. These students capitalised on available software and on procedures written by others. Group C students were labelled *machinists*, because they used their computers mostly as calculators and typewriters. Students in group D were labelled *perseverators* because of their tendency to use only procedures and programs written by others. They used these over and over again, and spent much time on the same task or activity. They liked drill and practice, particularly in spelling. The groups are described in more detail in the chapter. A small number of students in this study could not be classified on the basis of this scheme.

Other student characteristics reported in this chapter relate to the feelings and attitudes (including anxiety and enthusiasm) of students about computing, self image and confidence, knowledge of computers and computing, learning and problem solving, feelings of control, favourite computing activities and student expectations and perceptions of their teachers. These characteristics were assessed by means of structured interviews, questionnaires, observation and

objective tests. The final part of this chapter reports on an investigation of the students' own assessment of attitudes to and competence in computing in themselves and their peers. Of particular interest with respect to these findings are the students' criteria for assessment, i.e. how the students measured success.

In Chapter 6, Year 6 and Year 7 students are compared in their reactions to and knowledge of computing. Chapter 7 analyses the same attitude and knowledge variables but reports on individual differences between students in terms of ability and measured IQ. It also compares students who were judged by both their peers and their teachers as *doing particularly well in computing*, i.e. the R1 group, with those judged by the teachers as having difficulty in computing, i.e. the R5 group. The final section of Chapter 7 reports on an investigation of the programming *habits* of the students. Both the production of programs by the students and their understanding of programs written by others were assessed. Three of the tasks given to the students were taken from the published literature. This allowed at least a superficial comparison between the performance of the Australian students and a group of American students, even though the age of the students and the conditions under which they received instruction in computer programming differed in a number of respects. Chapter 7 closes with a brief demonstration of the interaction between students who were engaged in a collaborative programming assignment.

Chapter 8 discusses gender differences in relation to learning with computers. The literature argues that girls are often not given appropriate support and contexts for learning both about and with computers. Some of these findings were supported by our observations at Coombabah. Equal opportunity of boys and girls in access to computers is a problem in many classrooms, but obviously not in Coombabah, where each student has his/her own computer, but differences in participation are not just related to access to computers. We found that in the SUNRISE classrooms boys tended to dominate in discussions about programming procedures and in brainstorming activities. Possible factors contributing to the development of differential interest and achievement in computing, and particularly the development of gender based stereotypes are discussed in the chapter. Of particular interest are gender differences as they relate to computing in particular subject domains. In our empirical study, gender differences were not evident when the students first obtained their laptops; they developed over time and are stronger in the more experienced computer users (i.e. Year 7) than in the less experienced group.

PART IV: ASSISTING THE TEACHER (CHAPTERS 9 AND 10)

The chapters in this section deal with issues relating to the professional development of teachers who are teaching students with computers and make some suggestions relating to the contribution of teachers to the evaluation and development of educational computing software. The empirical study reported in this book was focused on the students learning with computers rather than on their teachers. As only five teachers were involved in the SUNRISE classrooms at Coombabah, the opinions of individuals and groups of teachers outside the project were sought to gather information for Chapters 9 and 10.

Chapter 9 reports on the aims, organisation and contents of staff development activities which could assist teachers with no or limited pre-service training in educational computing. Two likely broad aims for such staff development were identified:

1 to improve the skills and confidence of individual teachers in computer use; and
2 to persuade teachers to explore educational computing and to integrate computing into their teaching practice.

A number of problems relating to existing staff development offerings and the purposes of staff development are discussed. Consideration is given to questions such as whether staff development in educational computing should be voluntary or mandatory, whether incentives are necessary, the duration of courses, and how much training might be required.

Based on an extensive review of the staff development literature in Australia and overseas, published surveys of teachers teaching with computers and some published case studies, a set of conditions for staff development activities in personal computer use in schools was derived. These relate to the appropriateness, variability, incentives, maintenance, objectives, instructor, application and duration of professional development activities. The same literature review yielded a number of content topics and organisational features for professional development activities. A total of 60 Australian teachers, from three states, who teach with computers provided their reaction to the variables which had been derived from the literature review. On the basis of this a number of recommendations were formulated which were presented to the teachers in the SUNRISE classrooms at Coombabah. The teacher recommendations reported in Chapter 9 are thus based on the reactions of a much larger sample than would have been possible without the broader survey. These recommendations are not essentially different from those found in the recent overseas literature. Thus, they may well provide a basis for the planning of such activities in Australia.

The information provided in Chapter 10 is also based on a review of research on computer software and existing guides for its evaluation. The major message of this chapter is that software with improved pedagogical value can result if teachers play an expanded role in its design and development. Together with technical experts in computing and others, teachers should probably be involved in the development of instructional software and provide advice at all stages of development. Teachers are in an excellent position for identifying prerequisites for mastering the concepts and skills to be taught, as well as for deciding on the appropriate means for communicating the subject matter. They can help assure

that the software contains substance as far as its content is concerned, that it is error free, and that it engages appropriate thinking and problem solving skills on the part of the students. Teachers, because they are familiar with curricula in their subject domain, could assist in designing software which can be integrated or at least coordinated with other curriculum materials. The chapter concludes with an exemplary software evaluation procedure, developed by the New South Wales Department of Education (1985), which may be copied and used for educational purposes as long as the source is acknowledged.

Acknowledgements

The greatest single external contribution to this project has been that of Liddy Nevile, my former colleague at ACER, whose desire for a *school of the future* resulted in the idea of SUNRISE, a vision of making what the powerful tools of technology can provide available to *all*. She persuaded the Queensland Education Department to set up the Queensland SUNRISE Project and was instrumental in nourishing the slowly developing appreciation of the teachers for teaching and learning with computers.

I express my gratitude to the Queensland Department of Education, especially to Mr Robert Lenahan, Director Resource Management, Mr Laurie Vogler, Mr Jim Tunstall and Mr Ross Lever for allowing the project to go ahead during 1991. Mr Robin Ramsbotham and Mr Glen Finger, the Principal and Vice Principal of Coombabah Primary School and the teachers Jenny Betts, Karen Hallett, Barbara Macfarlane, Dave Mitchell and Zoe Schalch were very accommodating and helpful. I thank them sincerely for their tolerance and support. A big thank you goes to the Year 6 and Year 7 students who participated in the project.

Mr Greg Grimmett helped in many aspects of this project. Above all, he facilitated the communication process between Coombabah and Melbourne. I am most grateful for all his assistance and his valuable advice.

I gratefully acknowledge the assistance of the staff of the ACER library who provided an efficient and speedy service in meeting requests for literature searches and many inter-library loans, and the advice and assistance of Gloria Locock who gave the manuscript its final format.

Finally I express my appreciation to Professor Barry McGaw, Director of the ACER, for allowing me to take time to complete this project.

List of Figures

List of Tables

PART I

THEORETICAL FRAMEWORK

1

Personal Computers in the Pursuit of Educational Excellence

The overriding policy question today is no longer whether our schools have slipped into mediocrity or whether there is a crisis in primary and/or secondary education. Rather, the question is: What can be done to improve the current state of affairs? What can be done to turn Australia into a *clever country*? An obvious response is to capitalise on technological innovations, in particular on personal computers. However, three interrelated factors currently appear to be restricting the potential contribution of computers as generally accepted tools for learning and teaching from Kindergarten to Year 12: (1) a lack of knowledge about how best to educate teachers in the use of computers in education, (2) a lack of general curriculum objectives, and (3) a shortage of suitable instructional software which can be integrated with learning goals and local curricula. All of these factors are addressed in the following chapters.

Herbert Simon, who received the Nobel Memorial Prize in economics in 1978, referred to the invention of the computer as the *second industrial revolution* (Simon, 1987). He argues that, like the invention of the steam engine in the industrial revolution, the computer promises to dramatically increase the number and kinds of things we can do and to equally dramatically change the ways in which we do things. He reminds us, however, that it took the steam engine 150 years to have a pervasive influence on society, but computers have been around for less than 50 years. Simon also notes that the full impact of even these landmark inventions is tempered by and conditional upon other events, such as related inventions and accumulating experience resulting from their use.

COMPUTING IN SCHOOLS

For the computer to bring about a revolution in our schools and education more generally, this technological innovation must be accompanied by improvements in our understanding of the processes of learning and teaching and their implications for cognitive development, and by changes in the organisational structure of classrooms, schools, the curriculum and the broader learning contexts.

Computers have certainly transformed our society. As the products resulting from the industrial revolution amplified and boosted the physical power of humans, the computer revolution has the potential of increasing the power of the *mind*.

The first generation of computers began to transform society but not the schools. Problems of cost, size, and lack of appropriate software led to restricted access to computers. The invention of the personal computer has changed this. Despite some resistance from unprepared school personnel who had understandable doubts and were thus reluctant to accept the new technology, the easy availability of relatively low priced personal computers has brought about a second computer revolution during the past ten years, and this time schools are likely to be transformed.

Curiously, when politicians and people in the media speak of the *computer revolution* and its role in Australia's quest for educational excellence, learning and teaching are rarely mentioned, except perhaps to note that teachers might not be adequately prepared for the revolution and need to be provided with professional development. The focus tends to be on different kinds of hardware and its availability, on financial allocations by governments and on selected educational programs available at certain schools. From these news items one might gain the impression that the computer revolution has arrived and that a learning environment now exists in schools in which teachers and students make good use of computers, be it for drill and practice, for the development or remediation of basic skills, learning enrichment for the *gifted*, or computer literacy.

Unfortunately this impression does not accurately represent most environments in which teaching and learning with personal computers take place. The extent to which progress towards computer literacy and learning with computers is met actually depends on how well students and teachers are able to adapt, not to learning environments, which are close to *ideal*, but to reality, i.e. situations which contain relatively few resources and less than optimal curriculum materials.

Attempts to make the use of computers in schools more effective, therefore, must begin with the recognition that what is going on in educational computing in most schools currently is less than optimal. Progress can be made, however, for example by assuming that despite the limitations some classrooms in which personal computers are currently used for learning and teaching have produced highly successful students, and that the knowledge gained about learning and teaching decisions and practices, and recommendations for curriculum objectives, staff development and curriculum materials can help fill the current knowledge gap. Any attempts to fill this gap place us in a better position to identify what teachers and students need to know to use personal computers successfully in learning and instruction, what changes in classroom organisation and what improvements in resources, curriculum materials and material adaptation and development are required.

In the current debate regarding feasible roles of technology in education, and in particular advantages and disadvantages of using computers in the classroom, the machine itself is too often the starting point. Treated as the newest delivery system for teaching and learning, computer hardware and software, and other information technologies, become the main topic for discussion. A more productive debate of the uses of technology in education might result if the focus were moved to the processes of education, and the learner, rather than the physical components of the new technology.

> We need to be more concerned about the different ways students can learn when using the computer, the pressures on students now and in the future when they use computers, the learning processes underlying present and future software, the power of various kinds of feedback, and the psychology of the interaction between students and computers. (Hattie, 1988, pp. 1-2)

As educators we are concerned with improving both our basic understanding of children's learning and thinking, and the processes of education which can be expected to develop them. It is expected that this book will play at least a small part in improving this understanding with respect to the use of computers in the classroom.

A key component of learning and teaching is the processing of information. Students and teachers, like most other members of our society, collect, interpret and make decisions with respect to a vast array of information, some of which was gathered by earlier generations or by people in distant locations. Obviously, information processing has always been an aspect of learning, but the quantity of information now available to us is increasing exponentially. This is due largely to the availability of new technologies. At the same time we can make use of technology to help us handle all this new information. There is a long tradition of using technological aids such as printed matter, paper and pencil, an abacus or calculator, as tools to supplement and amplify human mental powers.

The computer is the latest addition to this range of tools. It is argued that by using the computer as a tool it becomes both an amplifier of human capabilities and a catalyst to intellectual development. This view of computing leads to more productive outcomes in the classroom than do those uses which turn the computer into a surrogate teacher. Obviously, the more effective uses of computers in education will require new patterns of interaction between students and teachers, changes in the social organisation of the classroom, the adaptation of curricula and alternative purposes and modes of student evaluation. All of these issues are addressed in this book.

COGNITIVE TECHNOLOGIES

Long before the arrival of computers, remarkable extensions of human intelligence were accomplished through the use of technical instruments. It is

taken as axiomatic that intelligence is not merely a quality of the *mind*, but a product of the relationship between mental structures and tools of the intellect provided by the environment, and more generally the culture (Bruner, 1966; Cole & Griffin, 1980; Luria, 1976, 1979; Olson, 1976, 1985; Olson & Bruner, 1974; Vygotsky, 1962, 1978). I shall refer to these tools as *cognitive technologies*. A cognitive technology is provided by any means that help transcend the limitations of the *mind*, such as restrictions in short-term memory, in activities such as thinking, learning and problem solving. The technologies which have tended to receive most attention in this respect are writing systems (e.g. Goody, 1977; Greenfield, 1984; Olson, 1977; Ong, 1982; Scribner & Cole, 1981) and systems of mathematical notation such as algebra or calculus.

Let us reflect on computers as cognitive technologies. Computers can store and dynamically manipulate symbols. It so happens that symbols appear also to serve as the *primitives* of human thought. Capable of real time interactions with human users, computer programming may well provide the most extraordinary cognitive technologies ever to be devised. Past experience with non-computer cognitive technologies may well help to inform and guide our definition of priorities for future uses of computers as cognitive technologies in education.

Cognitive technologies, such as written languages, are commonly thought of as *cultural amplifiers* of the intellect, to use Jerome Bruner's (1966, p. xii) influential phrase. They are viewed as cultural means for empowering human cognitive capacities. Greenfield (1966) observed that cultures with technologies such as written language will *push cognitive growth better, earlier and longer than others* (p. 654). As discussed in Chapter 2, we find similar predictions for computer technologies based on a widespread belief that computers will inevitably and profoundly amplify human mental power, and alter both what we do and how we do it.

The amplification metaphor for cognitive technologies has led to many research programs, particularly in relation to the cognitive consequences of literacy and schooling in the decades since Bruner and his colleagues published studies in cognitive growth (e.g. Bruner, 1964a, 1964b; Bruner & Tajfel, 1961; Greenfield, 1984; Olson, 1976; Scribner & Cole, 1981). This metaphor continued, for example in work on electronic technologies such as the prototype software systems for writing and mathematics in the form of *idea amplifiers*, *notebooks*, etc. and their uses for learning with computers, which are discussed in detail in later chapters of this volume.

While quantitative measures such as the efficiency and speed of problem solving, decision making and learning may truly describe changes that occur as a result of working with electronic tools, it can be shown that more profound changes -- as will be described in later chapters -- can be missed if we confine ourselves to the amplification perspective. Computing can do more than increase the efficiency, speed, reliability and comprehensiveness of our mental efforts: it can actually change the tasks problem solvers are faced with and thus alter the

cognitive processes involved in solving these tasks. Thus, computers can bring about change in the forms of thought itself.

Another tradition in the study of cognitive technologies can be characterised as cultural-historical. Influenced by the writings of Vico, Spinoza and Hegel, Marx and Engels developed a theory of society now described as historical or dialectical materialism. In this view, human nature rather than being a product of environmental forces, is of the person's own making and continually *becoming*. Humans are shaped through a dialectic of reciprocal influences: our productive activities change the world, thereby changing the ways in which the world can change us. By shaping nature and how our interactions with it are mediated, we change ourselves. As the biologist Stephen Jay Gould observes (1980), such *cultural evolution*, in contrast to Darwinian biological evolution, is defined by transmission of skills, knowledge and behaviour through learning across generations and has been our *nature-transcendent* innovation as a species.

From this cultural-historical perspective, labour is seen as the connecting link, i.e. the mediating factor, between humans and nature. By creating and using physical instruments and machinery which mediate in less and less direct ways our interactions with nature, we come to reshape human nature. Note how a change in the instruments of work (e.g. a plough rather than the hand) changes the functional organisation or system characteristics of people's fundamental relationship to work. Not only do humans accomplish their work faster, but what they do, i.e. their task, changes.

In efforts to integrate accounts of individual and cultural changes, the Soviet theorists Vygotsky (e.g. 1962, 1978) and Luria (e.g. 1976, 1979) generalised the historical materialism developed by Marx and Engels for physical instruments, and applied it to a historical analysis of symbolic tools such as written language which serve as instruments for redefining culture and human nature. Vygotsky (1978) recognised that

> the signs [symbols of language] act as an instrument of psychological activity in a manner analogous to the role of a tool in labour. (p. 52)

Using a Vygotskyan perspective, which stresses the functional reorganisation of cognition with the use of symbolic technologies, Cole & Griffin (1980) argued that the amplification metaphor has important shortcomings. Specifically they discussed how symbolic technologies qualitatively change the structure of the functional system for such mental activities as problem solving or memory. These fundamental changes are likely to go without being noticed if one thinks about cognitive technologies only with the amplification metaphor. Cole & Griffin highlighted how Luria enriched the term *function* for psychology. We are often inclined to assume one-to-one correspondences between functions and structures (e.g. planning is a function of the frontal cortex). In contrast, Luria speaks of the function of respiration not as the function of particular tissue, but as an entire functional system consisting of many components, such as the motor,

sensory and autonomic nervous systems. For Luria *functional systems are distinguished not only by the complexity of their structure, but also by the flexibility of the roles played by constituents* (Cole & Griffin, 1980, pp. 347-8).

In similar fashion Vygotsky saw shifts in functional systems of thinking as the sine qua non of developmental change:

> I have attempted to demonstrate that the course of child development is characterised by a radical alteration in the very structure of behaviour; at each new stage the child changes not only her response but carries out the response in new ways, drawing on new instruments of behaviour and replacing one psychological function by another. (Vygotsky, 1978, pp. 72-73)

By contrast, Cole & Griffin (1980) note how use of the term *amplify* means to make more powerful, and to amplify in the scientific sense. They suggest that amplification

> refers rather specifically to the intensification of a signal (acoustic, electronic) *which does not undergo change in its basic structure.* (p. 349)

As such, the amplification metaphor leads one to unidimensional, quantitative theorising about the effects of cognitive technologies. For example, the use of paper and pencil can be thought of as amplifying the power of a student's memory for a long list of words when only the outcome of the list length is considered. But it would be incorrect to go on to say that the mental process of remembering the words in the list, which led to a particular outcome, was amplified by the use of paper and pencil, because *remembering* in the two instances refers to two qualitatively different activities. The pencil does not amplify memory capacity, but it restructures the functional system for remembering, and thereby leads to a more powerful outcome (at least for the purpose of remembering more items).

Olson (1976), Ong (1982) and others argue in a similar way about restructurings of thinking process created through written language. For example, logical analysis of arguments for consistency/contradiction becomes possible because memory limitations for oral language are mitigated, and print (rather than oral narrative) provides a means to store and communicate cultural knowledge. It is important to remember that the possible restructurings of cognitive technologies are an empirical rather than a definitional matter. Cognitive restructurings are rarely predictable. They have emergent properties which come to be discovered only through their use. In this sense, as Dilthey (1976) urged, human history like evolution is a postdictive rather than a predictive discipline.

UNDERSTANDING THE IMPACT OF COMPUTERS

Society itself is rapidly and fundamentally changing in its structure and activities. Many of the changes are rooted in new ways of generating, storing, communicating and using information. We are shifting from the *industrial age* to the *information age*. In part this means that an increasing number of people are spending more and more of their time handling information.

Computers and related communication technologies are the visible signs of the information revolution. Hence, it has become important for all educated members of society to acquire basic *computer literacy*. This concept is discussed in the following chapters. Increasingly, parents, teachers and students believe that by learning about computers and by being able to use them, the students will be better prepared to survive and to enjoy economic well-being in the changing world. Educational computing has become the means for adapting schools to the new age.

What is it about computers that makes us so optimistic about their beneficial effects in the pursuit of educational excellence? The characteristics of the computer which most immediately account for its growing popularity and ubiquity are its ability to employ a wide range of symbols and to operate on symbolic expressions in powerful ways. In fact, these are the capabilities of computers which most closely correspond to human information processing. But computers not only employ a range of symbol systems, they differ from other media in the ways they can be used to structure and apply this information. The computer can be used to connect information based on the ways ideas are related to each other. Words can be linked to their definitions or to referent pictures. Concepts can be connected to examples for them. Theoretical principles can be linked to animated programs or video demonstrations. The computer's processing capability can be used to create procedural systems in which the information provided by the user determines what happens next. Such explicit representations of the relationships among information and symbolic expressions can serve as models for how knowledge can be related, structured, and used.

The emergence of the personal computer as an instructional tool has been surrounded by enormous publicity and speculation, which has tended to obscure many of the substantive issues surrounding its real and potential uses. Schools, parents and others in technologically advanced societies have shown themselves to be highly susceptible to the promise that the introduction of computers will provide the definitive answer to teaching and learning. In part, this susceptibility reflects one facet of our age, namely a fascination with machines and the belief that they might create progress and new frontiers. Every school strives for the latest and best in educational technology. However, the tendency to focus solely on the computer, rather than on the multitude of concurrent interactions with the learning environment, is a problem that faces teachers, curriculum planners and administrators alike. There is a growing realisation that new technologies will not

be easily integrated without a more definitive understanding of the interactional context into which they are being introduced.

Neither the features inherent in the machine nor the characteristics of the software will determine the influence of computers and computing in education. Rather, it is what people do with the machine that determines the influence of personal computers in any area of society. Hence the questions addressed in the study reported in this volume do not focus on the effects of computers on student learning, classroom structure, etc. The focus of the study is on what students do with the computer. How they *adapt the technology* for themselves and how they use it, rather than how the students themselves adapt *to* the technology.

HUMAN/MACHINE RELATIONSHIPS: ADAPTING THE TECHNOLOGY VERSUS ADAPTING *TO* IT

How could we view our relationship with the computer? What we will achieve with this technology will be determined by the uses which we can imagine for it. If we view the computer as we view a pencil, e.g. as a tool with which to produce a piece of writing, then we will obtain different results than if we view it in the way we do a wristwatch. This is the distinction between machines which work for us (e.g. motor cars, washing machines and other engines, watches, lights, and also computer programs designed for drill and practice) and those *with* which we work, i.e. tools (e.g. pencils, scissors, garden tools and word processors). We adapt *to* the machines which work for us and we adapt the machines and instruments which are our tools so that they best serve our purposes.

Human/machine relationships can also be thought of in terms of their degree of transparency, defined by the extent to which the machine becomes an extension of the human user or remains as separate, in psychological terms a *significant other*. Whereas a pencil or a spade is an extension to oneself, i.e. an addition which has been made to make it possible for a person to extend certain powers, to improve or make possible certain outcomes, other tools are not extensions as much as separate objects with their own purposes. They are agents rather than parts of the user.

Classroom computers have a primary effect which is transparent, or at least translucent, but it remains to be seen what the secondary effects will be. There will be motivational and social effects of educational computing, and effects on educational philosophy, but the most potent effect of computing is expected to relate to the development of students' powers of thinking.

Papert, one of the creators of the computer language *Logo*, and during the 1980s probably the leading exponent of the use of computer programming to expand students' intellectual power, proposed that computer programming environments in which children can deal with concepts, formerly regarded as too abstract for their developmental level, stimulate the development of important intellectual processes and can create conditions under which intellectual

processes may take root. Computer programming can make the abstract concrete and personal, and thus help children learn more effectively by making their thinking processes conscious. By programming the computer to do what the student wants it to do, students are forced to reflect on how they might deal with the task, and therefore on how they themselves think. Computer programming thus holds the promise of being an effective device for cognitive process instruction, i.e. teaching *how* rather than *what* to think.

There are many unanswered questions regarding the effects of computer programming experience on students' cognitive and personal development. Exploratory studies by the developers of Logo and others (e.g. Papert, 1980; Papert, Watt, diSessa & Weir, 1979; Clements & Gullo, 1984; Gorman & Bourne, 1983) indicate that even quite young students can learn to program and that they seem to profit intellectually. There is some evidence that programming can improve problem solving ability (e.g. Billings, 1983; Milner, 1973; Soloway, Lochhead & Clement, 1982). Other studies report considerable variability in skill levels attained by individual children, and that students' programming ability is often limited to specific contexts (e.g. Pea, Hawkins & Sheingold, 1983).

Neither pencils nor computers can be regarded as an *independent variable* which is introduced into a classroom, the *effects* of which can then be observed. The computer does not *cause* better or worse learning in mathematics or social studies. It does not *cause* more social interaction or less. The computer is not an agent but something which has become part of the learning environment and the total social environment in many different ways. No wonder that computers have been used in support of the most diametrically opposed theoretical approaches to learning and teaching. Computing can be used to make highly structured learning even more structured, and it can be used to make *open classrooms* more open. Above all, it can be used to increase the learner's self-directed exploration of learning tasks and problems. It can help students become better learners because it can provide individuals with explicit knowledge about their own learning and thinking processes.

Computing can provide a more personal relationship with many aspects of knowledge and thinking, because it is such a rich source of reference points for so many otherwise abstract ideas. Obviously the computer plays a role in learning and teaching even when it is not physically present. Papert points out:

> The computer in the head can often be a more effective aid to instruction than the computer on the desk. (Papert, 1987a, p. 182)

A popular argument by the early 1980s was that learning to program computers is more empowering for students than computer assisted instruction (CAI) or computer managed instruction (CMI), because in programming the user, rather than the computer, directs the interaction and is thus in control. In CAI and CMI the computer is in control.

Papert's book *Mindstorms: Children, computers and powerful ideas* ushered in a wave of enthusiasm about the powerful, cognitive benefits of teaching children to program computers. Another early advocate of computer literacy, Luehrman (1980) also argued that students should be taught how to use and control computers through programming them rather than being controlled by them as in CAI and CMI.

> [Computing] constitutes a new and fundamental intellectual resource. To use that resource as a mere delivery system for instruction, but not give a student computer instruction in how he might use the resource himself, has been the chief failure of the CAI or CMI efforts. What a loss of opportunity if the skills of [computing] were to be harnessed for the purpose of turning out masses of students who were unable to use computing. (pp. 133-4)

Both Papert and Luehrman view students as active learners. They envisaged the computer as a *tutee* to be programmed or a *tool* to be used by technically knowledgeable students to serve their own needs, terms popularised by Taylor (1980, pp. 1-4) to distinguish these computer uses from uses that envisage the computer as a mere delivery system.

If we want our students to be decision makers in the future, not passive recipients of technological accomplishments, we need to make sure that they develop empowering knowledge. We need to structure opportunities for them to question as well as to learn to use and understand technology. It is the development of the expectation and responsibility to question and inform policies, coupled with technical knowledge, which promises to be more empowering than technical knowledge alone. Watt (1982) describes this type of knowledge as the ability to understand the growing economic, social, and psychological impact of computers on individuals and groups within our society and on society as a whole.

> This includes the recognition that the computer applications embody particular social values and can have different kinds of impacts on different individuals and different segments of society. (Watt, 1982, p. 58)

Recognising that personal computers are dynamic components of a larger social system enables us to see the relationship between classroom culture, appreciation of computers, and learning and teaching as a mutually defining (rather than a unidirectional) relationship.

SOCIAL INTERACTION

Critics of the policy of providing students with personal computers often believe that computers will negatively affect social interaction in the classroom. Some parents have expressed fears that having unrestricted access to their own computers may result in children *spending many hours alone, sitting in front of the computer*, and will lead to isolation of the individual from his/her family and peers. An answer to this criticism is that there are *loners* of all ages who tend to withdraw from interaction with other people and focus their energies on themselves or their hobbies, e.g. their computer, their music, or their television set.

Studies on social interactions in classrooms containing computers indicate that computer presence actually increases the amount of interaction among the students (e.g. Emihovich & Miller, 1988a; Papert, 1987a) and our empirical findings in the SUNRISE classrooms at Coombabah strongly support these findings. Computing allows for dialogue between students, and projects that encourage cooperation. It allows students to create pieces of work or other outcomes about which they are excited and which they want to discuss with others. It facilitates new means of communication by exchanging and cooperating on files, sending messages by computer mail to recipients within and outside the classroom. In fact, one of the aims of the designers of Logo was to encourage communication between users.

> Logo programs are modular so that they can be borrowed and shared. Logo is also designed to make it as easy as possible to talk about how you made your program work -- what the bugs were, what the difficulties were, and how you solved them. Thus the content of actual computer work, even on what might seem like a very technical level such as designing a computer language, is a factor that can make for greater socialisation or greater isolation. (Papert, 1987a, p. 185)

Even in the rare classrooms where each student has his/her own computer, group work is likely to be the norm. Interaction between students during computer-based activities is regarded as beneficial, and teachers in such classrooms actually encourage pupils to talk, to reach group decisions and help each other with new procedures. For example, it has been shown that working in pairs or small groups at the word processor, even at the error correcting stage, can be highly beneficial. Students learn quickly from one another.

Lepper (1985) has commented on motivating aspects of computer use, and has pointed to the opportunities for an education that capitalises upon them. Papert refers to the sense of mastery over one's environment as a powerful motivator when children program in Logo, and Underwood & Underwood (1990) have found the same growth in self esteem, with consequent improvements in cognitive development when children work on databases. Self esteem may develop because children feel at the centre of their worlds. Motivation in such situations is due to the sense of positive power which comes from the ownership of a skill or knowledge, and is closely related to self esteem.

COMPUTING AS A CULTURAL COMPONENT IN THE LEARNING ENVIRONMENT

Papert (1987b) claimed that researchers who studied the effects of Logo on educational outcomes and cognitive processes were victims of what he called *technocentric thinking*. He defines this term as thinking which makes the properties of the machine the paramount feature of concern. Papert rejected the use of questions such as *What is the effect of the computer on cognitive development?* and *Does Logo work?* and suggested that these kinds of questions revealed a lack of understanding that 'the context for human development is always a culture, never an isolated technology' (p. 23).

The following quotations from Papert (1987b) provide some sense of his general argument:

> Developing a discourse is at the heart of developing a culture, and a more textured and knowledgeable discourse about Logo contributes to the *Logo culture*, the *computer culture*, and to the *learning culture* in its broadest sense. It sets the cultural context for personal learning. (p. 23)

> However, the finding as stated has no force whatsoever if you see Logo not as a treatment but as a cultural element -- something that can be powerful when it is integrated into a culture but is simply isolated technical knowledge when it is not. (p. 24)

> In a parallel way, I have sought to decenter the perception of the Logo experience. We are not looking at the effects of a technological object on an individual child, we are looking at the workings of a cultural process. (p. 29)

The second thrust of Papert's argument relates to methodological issues of how evidence should be collected to support the point that computer usage is embedded within a cultural context. Papert revealed a strong preference for anecdotes, single classroom case studies, and ethnographic research methods. He derided the use of experimental design as having little value because he believes the very nature of the design to be based on a one-to-one cause-effect relationship, which is ill-suited to revealing the multiplicity of factors that affect children's performance when using computers in the classroom.

Much of the debate about the effects of learning and teaching with computers in Australia and overseas assumes that a personal computer with a particular kind of software will have a specifiable and generalisable impact on students and teachers. Computing is regarded as a single factor of change introduced into a classroom which otherwise remains the same. In other words, the computer is perceived as an independent variable the effect of which can be controlled and quantified. In reality, computers in the classroom are far more than a treatment. They become inextricably intertwined not only with the way in which students might go about tasks, but with the whole context of learning and teaching.

Computers in the classroom alter the collaborative interaction and shared dialogue between students, and between students and teachers. The introduction

of computers changes the classroom culture. A fundamental feature of any attempts to evaluate the impact of this technology must thus be a focus on the dynamic interplay between learning processes, students, teachers and the learning context (cf. Rowe, 1991b). As noted above, it is not the features inherent in the machine but what students and teachers do with it that determines the effects of computing in schools.

Papert stresses the fact that children have access to programming and other computer applications in a multitude of ways, not just in school, and certainly not just as a treatment created by researchers who set out to compare performances between groups in controlled experiments. Although Papert may not encourage the use of Logo in experiments, he did imply that when it is used by teachers, positive outcomes are achieved. As evidence he stated that children who have been through Logo training obtained higher reading and attendance scores than children of similar background, not commenting on the causal relationship between differential use of Logo and different outcomes. These findings have been supported in more recent research.

Papert favours the use of Logo in natural experiments, while attributing the negative outcomes achieved in a controlled experimental study to the research designs used. While he did acknowledge that other factors could have played a role (e.g. parental motivation) in obtaining higher reading and attendance scores for the Logo groups, he uses this point to suggest that 'factors of this kind simply don't work one by one; they work as a web of mutually supporting, interacting processes' (Papert, 1987b, p. 26). In other words, he believes that controlled experiments cannot capture the web; they focus only on one factor at a time and miss the overall process of how the web evolved.

Of the three critics who responded publicly to Papert's (1987b) comments, only Pea (1987) addresses the cultural argument, though in a limited way. The other two (Becker, 1987; Walker, 1987) focused primarily on methodological issues which will be raised in later chapters of this volume. Pea emphasised that the research he and his colleagues conducted at the Bank Street College of Education, New York, was not limited to experimental studies. For example, the comprehensive, two-year case study conducted by Hawkins (1987), referred to in later chapters of this book, documented the experiences of three teachers who implemented Logo in their classrooms. The study revealed that Papert's claim of the value of Logo needed to be substantiated within the realities of classroom life. Even if Logo is conceived of as building a new culture of learning, it is still necessary to have more precise information about *how* it might affect children's performance in various domains, to determine differences in learning styles which children bring to computer usage and to consider these differences in designing instruction. Several other researchers besides Pea and his colleagues (e.g. Kull, 1986; Leron, 1985) have found that discovery learning or *messing about* with the computer, as Papert terms it, leaves many children unable to grasp the conceptual power of Logo to produce deep mathematical thinking in the way Papert originally envisaged it.

My view is that both sides have explained only a piece of the puzzle, because neither side has the appropriate conceptual starting point from which to begin the analysis of computer use in schools. Although Papert is on the right track in emphasising the cultural aspects of computer use, his framework is just too diffuse. It is almost impossible to move from his sweeping assertions to the level of students in the classroom, and then to locate classroom computer use within a wider social context. In contrast, Pea and his colleagues began their work by focusing on individual cognitive processes, and while these should be considered, a fuller account is needed of how cognitive processes are themselves embedded in social practice, perhaps along the lines suggested by Rogoff & Lave (1986). If computing in the classroom is a cultural component and a social practice, rather than a technological one, the crucial ingredient is people's experience with computing and not any inherent features of the hardware or software.

The development of computer literacy, for example, is a much more complex issue than simply learning how to use word processing, spreadsheets, databases and other programming capabilities of the computer. Several chapters of this book have been devoted to this topic. For the students of the 1990s and beyond who are preparing to live in our increasingly technological world, computer literacy makes up a large part of their *personal capital*. Obviously, this view raises issues of equity and access to computers, the same issues that are generic to the development of literacies in other areas such as reading, writing or mathematics.

COMPUTER LITERACY AS PERSONAL CAPITAL

Although there is little consensus over the term *computer literacy*, many of the definitions correspond to Papert's concept of *technocentrism* in that the effectiveness of the machine is considered primarily in terms of its use as a tool, either for personal efficiency or for carrying out instructions (given by others) more efficiently. Most definitions of computer literacy focus on the need for children to acquire expertise as programmers, for adults to use the computer as a tool to accomplish other goals, such as word processing, or for teachers to use the computer as a delivery system for CAI or CMI. In that none of these definitions view the computer as located within a complex social system like the classroom, they are analogous to traditional definitions of literacy as the simple acquisition of reading and writing skills.

As will be discussed in later chapters, the aims of computer literacy curricula range from those designed to rise general awareness of computers through skill in their use to the possession of broader understandings of the personal, educational, social, economic and political contexts and consequences of computer technology in society. The type and amount of knowledge one has about computers determines the potential of that knowledge to be socially and politically empowering.

Computer awareness is intended to provide the student with some general information about computer hardware, software, vocabulary, uses, history, and social impact. Students may or may not actually see or use a computer, but if they do, the use is usually limited to demonstrations. In their teaching of computer awareness, teachers rely on a variety of sources of information about computing, such as books, films and videos. Technical information and noncontroversial commercial applications are emphasised. Much of the information is produced and provided by the computer industry.

Another form of computer literacy emphasises *the ability to use computers*, where programming and applications dominate the curriculum. The *hands-on* approach is stressed, because the aim is to train students to *control the computer*. Programming, word processing, databases and spreadsheets are introduced in the elementary grades and more systematically elaborated in the middle school and high school mathematics and business education departments.

A third type of computer literacy is one which aims to have students make the computer part of themselves, and their personal work, as well as social environments. This type of computer literacy, which allows students to work with computers as malleable tools, rather than have computers work for them, is potentially most empowering, yet it appears to be the least discussed and experienced in schools. It introduces computing skills and knowledge within a broader social context, stresses the implications of computer technology and the empowering effects of such knowledge. The hands-on approach to teaching computing enables students to develop considerable computer specific skills, including graphic skills, and word processing. Students become confident in their interaction with the computer. But demystifying what is generally a user-friendly personal computer might contribute little to understanding the importance and potential of the technology for the individual. In fact, a narrow focus on the technical skills of using computers may even lead the students to a false sense of empowerment.

> The technical focus shifts attention away from social questions and portrays computers as something to learn rather than as something to think about . . . The computer is portrayed as friendly and accessible . . . and the user is encouraged to think that all computers, even those in large systems, are friendly and accessible. In this manner, computers are further mystified in the very act of demystification. (Noble, 1985, p. 72)

Students' technical knowledge needs to be accompanied by understanding and knowledge of who controls the direction of computing, for what purposes, for whose benefit and whose loss.

In the information age, the ability to access and use knowledge becomes a form of capital. It becomes a resource that allows those who have it to succeed in a society that depends increasingly on the manipulation of information rather than on just the production of goods and services. In the classic Marxist sense, the goods and services produced by human labour was the capital of society, but recently sociologists and economists have broadened that definition to include

other facets such as *intellectual* products as a form of capital as well. From this perspective, computer literacy can be described as personal as well as cultural capital. Treating computer literacy as a form of personal capital raises issues relating to opportunity of participation and equity of outcomes.

The personal capital of children who receive only remedial instruction in the use of computers, or are restricted to CAI and CMI, is severely reduced, since they have little opportunity to acquire a broad set of competencies applicable to a wide range of computing and more general problem solving and learning situations. In effect, they are *deskilled* (Apple, 1982), competent only in managing computer related tasks in which all the steps are prescribed. Just as the student's educational options are limited with only limited reading and writing skills, the same limitations occur with computer use where students can only follow directions and cannot adapt the computer for their own purposes. There is a large difference between students who experience the computer in terms of what it tells them to do and children who learn to view the computer as an interactive partner.

It is not too difficult to imagine a counter response to the argument presented above along the lines of *What's wrong with teaching students basic reading and writing skills on the computer?* No one would argue that reading and writing skills are not important. The key is that teachers can be teaching reading, writing and arithmetic at the same time as giving the students access to a sense of the potential and power of the technology that can fundamentally alter how they perceive themselves. In the *best* educational environments, computers will become an extension of the *mind* and they can allow students and teachers to see possibilities which they had previously hardly imagined.

One of the changes which has been apparent in classrooms where students are using computers as tools has been the subtle shift in the social order and power structure. The autonomy of the students increases. Students and teachers share experiences and become partners in learning. This can bring about

> an encounter between thought and reality, between desire and possibility, that takes places in the symbolic realm, and thereby vastly multiplies human capacity to process, analyse, criticise and re-invent experience. (Easton, 1989, p. 429)

This view supports Papert's (1980) original intent for developing Logo, a medium through which whole microworlds can be created on a computer and children can experiment with the simulation of multiple environments in endless ways. This sense of empowerment is especially important to children whose previous experience with school has consisted of one failure after another. As children become empowered in their quest for learning, the dynamics of the social learning environment change as well.

EMPOWERMENT THROUGH METACOGNITION

The term *empowerment* is used here in the sense of students' recognising that they have control over their ideas and thought processes, that they can access what they are thinking and then describe it to others. The psychological literature refers to this source of empowerment with the term *metacognition*, which means *thinking about one's own thinking*. It also refers to an individual's awareness and regulation of his/her cognitive processes and strategies (e.g. Brown, Bransford, Ferrara & Campione, 1983; Flavell, 1979, Rowe, 1988).

Metacognitive skills help students to monitor their strategy use during any cognitive enterprise, in accord with changing circumstances. The importance of metacognitive processes is that without them students find solving higher order problems almost impossible. These skills can be considered as another form of personal capital, one which many minority groups and/or children from deprived home backgrounds have been less successful in acquiring either in or out of school. The point to be stressed here is not that these students lack the cognitive capacity to acquire these skills, but that they have not been provided with the types of experiences that would help develop them.

In a series of studies investigating whether learning computer programming would affect the development of children's metacognitive abilities Emihovich and colleagues (Emihovich, 1989; Emihovich & Miller, 1988a, 1988b, 1988c; Miller & Emihovich, 1986) wanted to learn whether teaching children (ranging from preschool to grade 4) in urban and rural schools how to program a computer would help them become more aware of their thinking processes, in terms of how they planned and executed solutions to problems they generated themselves. Logo was used because it enabled students to be involved in programming activities. To make the *turtle* (i.e. the cursor) move, the students type in a set of commands indicating direction, along with a number of spaces they want the turtle to move. For example, *forward 50* or *(FD 50)* would result in a line on the screen from one spot to another. In short, what the students see on the screen would be a graphic representation of the image of the move they had planned in their minds.

Most of the children the above investigators worked with had obtained low scores on standardised psychometric and achievement tests. They were representative of the children who are typically denied experiences to *stretch their minds*, because of a concern by teachers over their lack of basic skills. Despite these perceived deficiencies, however, these children succeeded over time in learning to combine commands to create very complex designs. The investigators believe that this programming experience became an empowering one for the children for two reasons:

1 It provided the children with an opportunity to excel in a task which was cognitively demanding. They worked on tasks which, as a result of their low test scores, these students would not have been considered competent enough

to attempt in most schools. Although in one study the investigators were able to show a relationship between Logo programming and improved performance on a standardised mathematics test, that was not the major purpose of the research. The investigators wanted to prove that certain children should not be denied access to computer programming and computer literacy simply on the basis of previous test scores. Access to computer literacy, as demonstrated in these experiments, is a strong source of empowerment.

2 The investigators did not view computer literacy in the sense of students becoming expert programmers, but in the sense of children becoming aware of the fact that their ways of thinking, speaking, and writing could be mapped onto a powerful piece of technology which allows them to have access to the content of their thoughts. Using the turtle allowed the children to see the connection between what they had envisaged in their minds and what they actually drew on the screen. As described by a 6th grade student in the SUNRISE classroom at Coombabah: *It's like drawing things straight from my mind. Like, whatever you think you draw.*

Olson (1985) has suggested that *to be intelligent in the society of computer users is to be skilled in making one's meaning explicit* (p. 7). Through programming students become more aware of the need to be explicit about the way they communicate with the *turtle*, i.e. exactly how they go about tasks. This awareness will lead our students to become more independent and self-directed in their learning, and to take control of their own motivational, problem solving and learning efforts.

THE ROLE OF TEACHERS

As a result of the new educational technologies the work of teaching and the role of teachers are likely to change with respect to curriculum content, classroom management and student assessment. The computer is a powerful tool and tool box. It is an instrument which facilitates the acquisition of knowledge and skills in active ways. The student is provided with an environment which is conducive to exploratory learning. Learning through exploration puts high cognitive demands on students. At times this may result in inefficient and ineffective learning strategies, where learners flounder and do not use the opportunities the classroom environment offers. This is why support is needed if learning with computers is to be effective. Most of this support should be given by the human teacher, although some can be derived directly from the computer software.

In describing children's experiences with computing we must not lose sight of the fact that successful acquisition of computer literacy depends not only on changes in individual cognitive processes but to a large extent on the social practices surrounding how the instruction is presented. The way computing is taught in the SUNRISE classrooms at Coombabah, while not intentionally based

on the work of Vygotsky (1978), could certainly be reconciled with his theory. Vygotsky emphasised the role of social interaction in the modification and development of metacognitive skills in children. Vygotsky's concept of *the zone of proximal development* was defined as

> the distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem solving under adult guidance or in collaboration with more capable peers. (p. 68)

In the above noted studies by Emihovich and her colleagues, adult guidance was provided by tutors who helped the children understand the processes involved in programming in order to bridge the gap between the child's understanding of the task, what was accomplished and the outcomes which were expected in the classroom. This role is played by the teachers. Skilled and sensitive teachers have always known that they must provide the *scaffold* to initially assist children to understand what is required of them. A more critical aspect of this concept, however, is that by providing a particular type of instruction, which might be called mediated instruction, and which is often termed mediated learning, it is the teacher who is able to help students to extend their performances beyond the levels they could reach independently. In short, only teachers (and in some cases more experienced peers) can help students cross the zone between what they know and what teachers know they could do with assistance.

Mediated instruction does not mean the teacher is there to direct children to execute specified tasks. Rather the teacher serves as a facilitator in helping the child reflect and think about how he/she arrived at an answer, or how they planned and carried out their ideas using the turtle. In this framework the role of the teacher is critical, especially with young children, because the teacher is the one who initially helps the student construct meaning out of an activity.

The need for collaborative interaction between students and teachers, i.e. the need for a shared dialogue about learning, is one piece that was missing in Papert's early writings about the use of Logo. He relied too much on a *stand alone* model of human/computer interaction derived from the ideas of artificial intelligence (AI). In the AI models, children learn all they need to know from highly interactive machines simply by engaging in the creation of *microworlds*.

Although the quotations cited earlier from Papert's (1987b) article suggest that he recognised that Logo must be seen as a cultural process, his precise interpretation of the meaning of this is not clear. In contrast, it is clear that he saw the teacher as having a peripheral role, if any at all. Papert is not alone in this view. After reviewing the literature, Olson (1988) noted that most writing about computers in the classroom is dominated by the AI view that teachers are a hindrance to technological progress, and that the *mindlessness* of traditional teaching will be replaced by students engaged in stimulating problem solving activities alone at the computer. Olson disagrees with this proposition:

> Talk of thinking skills, of autonomy, of discipline, does not consider how school subjects become meaningful for children. Is not the teacher the very resource students need to construct meaning, because teachers are intelligent? Teachers really can talk back. Furthermore, school subjects provide teachers with tools for creating meaningful contexts for learning. Is talk of decoupling learning from teachers simply to talk about making learning less meaningful? These are the issues that emerge from thinking that teachers can be supplanted and school subjects transcended. (Olson, 1988, p. 9)

The computer is no substitute for the individual, experienced teacher. Rather, it offers a number of new teaching opportunities. In order to be effective, learning with computers requires the presence of a teacher to monitor the performance of individual students and provide both directive and non-directive support.

Approaches to professional training which view the teacher solely as a bystander, as a technician or as a mere consumer of curricula that others design, are likely to be completely inadequate for the preparation of teachers for classrooms in which students learn with computers. More importantly, such approaches are unlikely to provide teachers with a significant professional role in shaping the future technological transformations of schools, which will be adopted by future teachers only in so far as they are meaningful and integral to their teaching situations. Teachers must be encouraged to become partners in the creative enterprise of curriculum (including software) development.

As is stressed in later chapters of this book, ideally, teachers will be the central participants in and builders of the future of technology in education, not solely the recipients of decisions made by others, either in the area of training or tool design. Teachers must be supported and encouraged to adapt educational computing to their own and their students' purposes, to explore the ways in which technologies can alter what happens in the classroom, and to share what they do and what works with other teachers. Their influence should be felt in what is produced and marketed for schools during the process of software and curriculum development, not after.

Professional development programs must support teachers to shape and engage in experiments with technology in education. Some of the most imaginative and successful uses of computers in classrooms today have come from teachers who were willing to redesign learning activities to take advantage of the technology, or who have discovered new dimensions in the technology that cou ld be shaped and revised for use in the classroom.

CONCLUSION

The use of personal computers in the pursuit of educational excellence in schools is a new territory, which means that there is too little background in the literature (both empirical and theoretical) to give answers to all the issues and questions raised here and in the body of this book. This should not discourage the reader, but rather become a compelling motive for deep reflection and consideration of

the philosophical and pedagogical assumptions, educational goals and classroom practices in the use of personal computers in schools.

As will become evident, the experiment conducted in the SUNRISE classrooms at Coombabah Primary School has shown that personal computers can be successfully integrated into classrooms. The observations made during the 1991 school year give some powerful insights into what can occur when teachers teach students who have their own computers. This book discusses theoretical and practical issues which arise when computers are introduced, and opportunities offered to students and teachers through this technological innovation.

However, predicting the long-term outcomes of new developments is difficult in any field. During the past five years we have certainly witnessed important changes in the use of computers in schools, but what does this mean in terms of the potential implications of educational computing? We recognise that the computer is an enabling tool which facilitates the execution of many routine and non-routine processes in creative production, problem solving and learning. Computers can facilitate the achievement of many valued learning goals, but their role is not a simple one. It is not sufficient to buy a personal computer and software, place the student in front of the computer and have the computer work its *magic*. Their is no *magic* in computing: rather, because of its interactive aspects the computer allows the student a more active part in constructing knowledge than paper and pencil could. Understanding the impact of personal computers in classrooms means understanding the complex system of interactive relationships between people, situations, tasks, social and cultural processes, and the learning context of which the computer is an intregral part.

As Simon (1987) pointed out, the success of a major innovation is dependent on and subject to the occurrence of a variety of concomitant events and conditions. Consequently, the impact of the personal computer on classroom learning and teaching cannot be assessed, or even considered, in isolation. Computing in the classroom cannot be disentangled from the cognitive, social and personal demands of the curriculum goals and instructional tasks which teachers set for their students, or from the interests, motivations, skills, knowledge, abilities and difficulties which students bring to the learning situation. Understanding the impact of computers in education means understanding this complex dynamic system of variables as a whole.

2

Cognitive Effects for Students

The use of computers in classrooms in general, and projects such as the one conducted in the SUNRISE classrooms at Coombabah in particular, raise issues of the following kind: What exactly does computer technology offer the processes of education? What is unique about the function of the computer as a tool of the intellect? How does or should the technology, as used in society, influence what is done in schools with computers? How might information technology redefine the very possibilities of learning and teaching? How can our practices and our research contribute to the development of students and to effective curriculum design? Or just simply: What is the use of computers in the classroom? Some of these questions are analysed in this chapter and the next.

Computers are influencing, in a very fundamental way, the traditional organisation and definition of many jobs and workplaces in industry, business and the public sector. Workplaces are undergoing physical transformations as well as changes in job specifications and job category distribution (e.g. Noyelle, 1984; Cyert & Mowery, 1987), and new social interaction patterns are brought about by the electronic environment. Specific performance demands are also shifting, for example from manual to sensory discrimination, and towards the processing of text, suggesting that technology is affecting basic psychological activity (e.g. Martin & Scribner, 1991). The same is true for the classroom, i.e. the predominant work environment for students and teachers.

A general view is that the availability and use of computers increases people's productivity and efficiency. Computers can extend, supplement and thus boost human performance. An alternative implication of computing is that it can provide a *box of tools* which are actually capable of changing the characteristics of problems and learning tasks, and hence somehow lead to a restructuring of the processes of learning and problem solving more generally. This latter attribute of computing might well contribute new visions of the potential cognitive benefits of the technology.

The question arises: Might the learning of computing have positive effects on students' problem solving and reasoning? To answer this question, two important and related questions need to be investigated: Which components of instruction in computing could result in improvements in problem solving and reasoning skills? Which sequential steps of mastery of computing skills might lead to such improvements? Some reflections on the relationship of problem solving and

computer programming, and a discussion of an initial set of steps towards an elementary level of mastery in programming, described in this chapter, are a preliminary attempt to deal with these questions.

Early Research

Two major influences appear to have contributed to the belief that programming may spontaneously discipline thinking. The first is from artificial intelligence, where constructing programs that model the complexities of human cognition is viewed as a way of understanding these complexities of human behaviour. The contention is that in explicitly teaching someone (or the computer) something, one learns more about one's own thinking. Papert (1972a) postulates that through programming students would learn about problem solving processes by means of the necessarily explicit nature of programming, i.e. as they articulate assumptions and precisely specify steps in their problem solving during programming. The second influence is the widespread assimilation by educationists of constructivist epistemologies of learning, most familiar through the work of Piaget (e.g. 1970 1972, 1973). Papert (1972a, 1980) has been a strong advocate of the Piagetian theory of knowledge acquisition through self-guided problem solving experiences, and has extensively influenced views relating to the benefits of learning to program through *learning without curriculum* in *a process that takes place without deliberate or organised teaching* (Papert, 1980, pp. 27-28). It should be noted here that Piaget is not advocating the elimination of organised teaching in schools.

For a long time there appeared to be a considerable gap between the rhetoric about the overt and latent benefits of educational computing and classroom reality. Grand claims have been made about the potentially positive implications of educational computing as a tool of the intellect. Some of these claims are supported by evidence gained from small scale studies conducted under almost ideal circumstances, such as teaching by enthusiastic experts who have generous resources. More substantial claims for the effects of learning programming on thinking are exemplified in the writings of Papert & Feurzeig (e.g. Feurzeig, Horwitz & Nickerson, 1981; Feurzeig, Papert, Bloom, Grant & Solomon, 1969; Goldstein & Papert, 1977; Papert, 1972a, 1972b, 1980; Papert, Watt, diSessa & Weir 1979) concerning the Logo programming language, although such claims are not unique to Logo (cf. Minsky, 1970; Nickerson, 1982).

Ross & Howe (1981) have translated Feurzeig, Papert, Bloom, Grant & Solomon's (1969) four claims for the expected cognitive benefits of the development of mathematical thought to the learning of computer programming and proposed:

that programming provides some justification for and illustration of, formal mathematical rigor;
2 that programming encourages children to study mathematics through exploratory activity;
3 that programming gives key insights into certain mathematical concepts; and
4 that programming provides a context for problem solving, and a language with which the pupils may describe their own problem solving. (Ross & Howe, 1981, p. 143)

Papert (1972b) argued for claims (2) and (4) by noting that writing programs of turtle geometry is

> a new piece of mathematics with the property that it allows clear discussion and simple models of heuristics [such as debugging] that are foggy and confusing for beginners when presented in the context of more traditional elementary mathematics. (p. 252)

He provides anecdotes of children making spontaneous discoveries relating to phenomena such as the effects of varying numerical inputs to a procedure for drawing a spiral on the shape of the spiral. He concludes that learning to make these small discoveries brings the child closer to mathematical thinking than being taught new mathematical concepts. Papert (1980) discusses the pedagogy surrounding Logo and he argues that cognitive benefits will emerge from taking *powerful ideas* inherent in programming, such as recursion and the concept of a variable, in mind-size bites.

Feurzeig, Horwitz & Nickerson (1981) provide an extensive set of cognitive outcomes expected from learning to program. They argue that the teaching of concepts related to programming can be used to provide a natural foundation for the teaching of mathematics, and indeed for logical and rigorous thinking more generally.

More Recent Findings

There is a considerable body of opinion which focuses on the real contributions computers can make to education, tempered with an awareness of the barriers to educational change which are met by all educational innovations, and the likely slow rate of progress.

In a synthesis of the results of thirteen quantitative reviews of research into the benefits of computer-based instruction, Niemiec & Walberg (1991) covered more than 250 individual research studies and showed the typical and average effect of computer-based instruction to be that it raises learning outcomes by .42 of a standard deviation. Although differential effects were noted by the authors of this analysis, they found that the overall effect of computer-based instruction placed the students in computer-based instruction at approximately the 66th percentile of the control group distribution.

Other relevant research projects, many of them conducted at Bank Street College, New York, include studies of the development of problem solving and

planning skills in Logo programming (e.g. Kurland & Pea, 1985; Pea & Kurland, 1984; Pea, Kurland & Hawkins, 1985), concern with cognitive demands and consequences of learning programming (e.g. Clement, 1984; Kurland, 1984; Kurland, Clement, Mawby & Pea, 1987), classroom uses of software such as database management systems and word processors (e.g. Freeman, Hawkins & Char, 1984), investigations into how teachers' interpretive frameworks for software are linked to how they reorganise classroom learning with new technologies (e.g. Hawkins, 1983; Hawkins & Sheingold, 1987 ; Sheingold, Hawkins & Char, 1984), and formative research aimed at the creation of computer or multi-media instructional packages of an open-ended nature for student learning in mathematics, science, languages and technology (e.g. Char, Hawkins, Wootten, Sheingold & Roberts, 1983). In these studies the term *problem solving* has a broader and deeper meaning than its often restrictive educational association with mathematics implies. Computer environments make it possible for students to experience some of the deeper ideas that underlie a correct understanding of what human problem solving entails.

Some of the claims made in the literature suggesting that learning to program can be expected to bring about fundamental changes in thought are summarised below. Expected improvements include:

- rigorous thinking, the development of precise expression, and a recognised need to make assumptions explicit (because programs operate on the basis of specific algorithms);
- an understanding of general concepts such as a formal procedure, variable function, and transformation (as these are used in programming);
- the general idea that one can invent small procedures as building blocks for the gradual construction of solutions to large problems (as programs are composed of procedures, templates, etc.)
- greater facility with heuristics, explicit approaches to problems useful for solving problems in any domain, such as planning, finding a related problem, solving the problem by decomposing it into parts, etc. (because programming provides highly motivating models for the use of heuristic concepts);
- the general idea that debugging of errors is a constructive planning activity applicable to any kind of problem solving (because it is so integral to the interactive nature of the task of getting programs to run as intended);
- generally enhanced literacy and metacognitive awareness with respect to the processes involved in solving problems (due to the practice of discussing the process of problem solving in programming by means of the language of programming concepts);
- enhanced recognition for domains beyond programming that there is rarely a single best way to achieve a goal, and an understanding that different ways have comparative costs and benefits with respect to specific goals (learning to distinguish between process and product).

Clements & Gullo (1984) advance the following exploratory hypotheses in relation to possible cognitive benefits of computer programming:

> In Logo programming children invent, construct and modify their own projects; therefore, Logo programming might facilitate divergent thin ing.
> 2 Because Logo is designed to encourage children to reflect on how they think, programming should lead them to develop metacognitive abilities, especially the ability to realize when they do and do not understand instructions.
> 3 Similarly, Logo programming may develop reflectivity in children as they think about their errors and how to correct them.
> 4 If computer programming can allow children to master ideas formerly thought too abstract for their developmental level, it may accelerate cognitive development, including operational competence.
> 5 Finally, because Logo programming involves giving explicit spatial commands, it should increase children's ability to describe directions from their own and others' perspectives. (Clements & Gullo, 1984, p. 1052)

It is possible, of course, that any benefits derived from computer programming can be attributed to the interactive nature of computing, rather than to the programming activities *per se*. It would therefore be necessary to provide a control group with computer experience not involving computer programming. Such experience might consist of using software containing word processing, ready made databases and spreadsheets, or they might consist of computer assisted instruction (CAI). Clements & Gullo (1984) set up such a study.

CAI has its roots in programmed learning and thus has a strong connection to the behaviourist tradition. Emerging from three themes of learning theory, i.e. individualisation, behavioural objectives and educational technology, many CAI programs employ the approach of programmed learning. Thus, they share the following characteristics: (a) they store a sequenced series of steps, often providing alternative learning paths for individuals, (b) they offer independent pacing for individuals, (c) they provide students with controlled contingent re-enforcement, and (d) they can evaluate performance quickly and accurately to provide feedback on the degree of mastery.

Clements & Gullo (1984) compared a Logo group and a CAI (control) group of students and the effects of each on 3rd grade children's cognitive style (including reflectivity, divergent thinking, etc.), metacognitive ability, operational competence, and overall cognitive development. The two groups did not differ significantly prior to treatment in the language and cognitive domains as measured by the *Peabody Picture Vocabulary Test-Revised Edition, PPVT-R* (Dunn & Dunn, 1981). The study revealed significant pre- to post-test differences on the *Tests of Creative Thinking* (Torrance, 1974) for the Logo group on *fluency* and *originality*, as well as on the overall *divergent thinking* score, while no significant differences were found for the CAI group. They also found that in the Logo group the latency time increased and the number of errors decreased. The Logo group significantly outperformed the CAI group on two metacognitive tasks. The ability to monitor one's own thinking and realise when one does not understand are likely to be influenced by programming environments in which problems and solution processes are brought to an explicit level of awareness, monitoring and subsequent modification. Through consistent feedback in the form of visual representation of the procedures and sequences of their own thinking processes, the Logo students may have learnt how to monitor these processes.

The scores on a test of describing directions were similarly affected. The Logo group, being face to face with the turtle, practised orienting themselves to the turtle's visual perspective. This skill is a prerequisite to the successful completion of the *Describing Directions Test* in the *Kaufman Assessment Battery for Children, K-ABC* (Kaufman & Kaufman, 1983). The directions of this test involve left-to-right and top-to-bottom reversals. Research has shown that with practice children improve on visual perspective taking (Flavell, 1977; Donaldson, 1978). No difference was found between the groups in two areas of cognitive development -- operational competence (classification and seriation) and other aspects measured by the *McCarthy Screening Test* (McCarthy, 1978).

Linn & Dalbey (1985) showed in a study involving over 500 pre-college students in 17 classes that the form of instruction, the access to computers, and the ability of the student influence outcomes from programming instruction. Specifically, *exemplary instruction* was found to move students further towards mastery of component skills than do less effective or as described by Linn & Dalbey, *typical* methods of instruction. Furthermore, both access to computers and the general ability of the students were found to be related to progress in *typical* classrooms. In *exemplary* classrooms, for medium and high ability students, neither ability nor computer access outside of school were found to be related to programming performance.

As noted previously, there is a widely held belief that computers will influence how effectively we accomplish traditional tasks, supplementing or extending and thus boosting human cognitive capabilities, on the assumption that the tasks stay fundamentally the same. The central point made in the context of the Coombabah classrooms and other projects where children learn *with* computers, rather than *about* computers, is quite different. Here, the primary role of computing is seen as one of changing the tasks and what teachers and students do, thus reorganising or restructuring tasks and possibly mental functioning, not only by extending and supplementing it. The predominant use of computers in schools today is with software that aims to make long-familiar drill and practice activities, especially in mathematics and language, more efficient and effective. Why not focus instead on using software as tools to support and restructure the student's thinking?

COMPUTER PROGRAMMING AND PROBLEM SOLVING

At the core of computer programming is that set of activities, involved in the development of a re-usable product, consisting of a series of instructions, which

make the computer accomplish a given task. As is the case in more general theories of problem solving, cognitive studies of programming reveal a set of distinctive mental activities that occur as computer programs are developed. Some of these activities are involved throughout program development, irrespective of whether the programmer is an expert or a novice, because they constitute recursive phases of the problem solving process in any context and theory of problem solving (e.g. Heller & Greeno, 1979; Newell & Simon, 1972; Polya, 1957; Raaheim, 1974; Rowe, 1985, 1991a). They may be summarised as follows:

1 understanding and defining the programming problem,
2 planning or designing a programming solution,
3 writing the programming code that implements the plan,
4 comprehension of the written program, and
5 program debugging.

Planning

One of the claims made about the positive effects of programming on thinking has been in the area of planning. The assumption is that programming experience will result in greater facility with *heuristics*, explicit approaches to problems useful for solving problems in *any* domain. It is possible, however, that students who can think logically, plan, and have acquired reasonable problem solving heuristics develop programming skills.

One may raise the objection that it is possible to bypass planning in program development, i.e. one might first make an initial reading of the problem and then compose code at the keyboard to accomplish the task. Although planning as one proceeds is certainly possible in the production of some programs, it seems likely that such attempts might create problems for the inexperienced programmer. While expert programmers can draw on their knowledge of a vast range of plans when creating a new program, the novice programmer has neither the sophisticated understanding of programming code nor the experience of devising the successful programming schemas which are necessary for engaging in planning as they proceed.

Planning can be characterised as a process of revision. As a consequence of considering alternatives, effective planners revise their plans. They alternate between top-down planning strategies, which create a plan from successively refining the goal into a sequence of subgoals for achievement in sequence and bottom-up planning strategies, which note the emerging properties of the plan or the planning environment and add data-driven decisions to the plan throughout its creation (e.g. Hayes-Roth & Hayes-Roth, 1979; Pea, 1982).

In the Coombabah project, most children appeared to do little planning in their programming work. Planning before writing an essay was insisted upon by the

teachers, pre-planning before programming was mentioned by some teachers but not insisted upon, and explicit pre-planning aids (e.g. worksheets) were not provided. Students appeared to write and revise their code in terms of the immediate effects that commands and sequences of commands produced.

Pea, Kurland & Hawkins (1987) found that students who had spent a year programming in Logo did not differ on various developmental comparisons of the effectiveness of their plans and their planning processes from students who had not learnt to program computers. They concluded that learning thinking skills and how to plan well are not intrinsically guaranteed by the Logo programming environment. Rather, the development of planning skills must be supported by teachers who, tacitly or explicitly, know how to foster the growth of such skills through judicious use of examples, student projects and direct instruction. This is in contrast to the Logo instructional environment which Papert (1980) offers to educators, which is devoid of curriculum, and lacks an account of how the technology can be used as a tool to stimulate students' thinking about such powerful ideas as planning and problem decomposition.

> Teachers are told not to teach, but are not told what to substitute for teaching. Thinking skills curricula are beginning to appear, but teachers cannot be expected to induce lessons about the power of planning methods from self-generated product-oriented programming projects. (Pea, Kurland & Hawkins, 1987, p. 196)

Cohen & Feigenbaum (1982) define problem solving as a process in which a sequence of actions is developed to achieve some goal. These authors understand planning to mean deciding upon a course of actions prior to initiating the actions. A plan may consist of an unordered list of goals or an ordered set of goals (first do this, then do that, etc.). In a sense most of the research in artificial intelligence can be subsumed under this broad concept of problem solving research.

Basic Skills and Their Application

In the development of human minds two broad classes of activity are of particular importance: (1) activities which serve to equip children with a toolkit of basic mental skills, and (2) activities which require the application of those skills in generalised problem solving. Whereas the toolkit of the educated student will include specific arithmetic and language skills as well as more general cognitive skills, such as the ability to question and to categorise, the more widely based activity of problem solving requires the manipulation of information through the use of combinations of these and other skills. The overall aim is to provide students with a possibility of improving their thinking abilities. This is achieved by providing them with the basic skills, i.e. the cognitive toolkit, and with experiences in the use of different combinations of the components of their toolkit in problem solving exercises.

Because programming involves solving problems, it is generally assumed that learning to program, at least superficially, leads to improved problem solving skills in students. Hence, in most situations where programming is taught, teachers expect at the same time to improve students' problem solving and reasoning skills. In his seminal work *Mindstorms* Papert writes:

> In my vision, the child programs the computer and, in doing so, both acquires a sense of mastery over a piece of the most modern and powerful technology and establishes an intimate contact with some of the deepest ideas from science, from mathematics, and from the art of intellectual model building. (Papert, 1980, p. 5)

In contrast, outcomes from actual programming instruction often fall short of these expectations (e.g. Pea & Kurland, 1983; Dalbey & Linn, 1985). Furthermore, cognitive research has shown that problem solving abilities appear to be much more subject or discipline specific than had first been thought (e.g. Larkin, McDermott, Simon & Simon, 1980; Linn, 1985a; Resnick, 1983). Thus, the mechanisms that might lead to generalisation of problem solving ability from one discipline to another require specification.

Programming a computer is a form of problem solving. Superficially, at least, students who learn to program acquire reasoning skills and learn about problem solving. Furthermore, certain features of the computer learning environment demand rather complete and accurate solutions to problems. For example, computers require precise input because they only respond to a limited set of commands. Programmers must decompose complex problems into subproblems and then generate sets of step by step instructions to solve the subproblems. In addition, the learning environment of computer programming is interactive in that by testing the problem solution, the problem solver can receive feedback about how effective the solution is and can use this information to modify the solution. Thus, writing a program requires the student to explicitly use some potentially powerful problem solving skills.

Programming is also compelling as a vehicle for teaching problem solving because many students are highly motivated to use the computer (e.g. Lepper, 1985). Programming sessions frequently allow students to try out procedures, even when students have little idea of how they will use them. Lepper notes that, among other things, the precision and interaction of the environment as well as the potential for challenge and fantasy appeal to students.

Computer programming, therefore, seems ideal for encouraging problem solving. But will the problem solving skills generalise to other areas of learning? Cognitive research has shown that learning is much more discipline-specific than had first been thought (e.g. Larkin, McDermott, Simon & Simon, 1980; Resnick, 1983; Scribner & Cole, 1981). When students learn to solve physics problems they learn about physics but not necessarily about problem solving in general. Thus, when students learn to solve programming problems they may not automatically or necessarily learn to solve other problems.

Acquiring a Toolkit of Knowledge and Skills

The perception of a shrinking world is brought about largely by technological change. Although it is not a new phenomenon, the exponential increase in information and the associated accelerating rate of change in the latter half of the twentieth century has increased the feeling of a shrinking world. Caught in an information deluge from around the world, we must look for new tools to help us cope. We need bigger memory stores, faster and more accurate retrieval systems and, above all, the ability to sift and sort the information before we *drown* in it. The computer is a tool designed to extend our overstretched human capabilities.

It can be argued that our future economic success depends on the degree to which children are taught to be sufficiently flexible and adaptable in their thinking and actions in order to handle the pace of change brought about by the information age. Gagne (1970) has argued that the most important things learnt in schools are intellectual skills, not pieces of verbalisable knowledge. This is the distinction between declarative knowledge, i.e. to *know what,* and procedural knowledge, i.e. to *know how,* both of which are central to the idea of a generalised cognitive toolkit. However, it is the acquisition of procedural knowledge which we should be looking for in classrooms that are dedicated to flexible problem solving, even though Longworth (1981) paints quite a gloomy picture. He considers that what today's children learn at school, at best, has a useful life of half a generation, while at worst it is obsolete as it is being taught. Competence with a variety of new technologies and their applications will become more useful as we enter the *hole in the wall* society in which not only cash but also information will be dispensed only to those who know how to gain access to it.

There is strong evidence that the current emphasis in classrooms is upon factual knowledge combined with a very limited structural organisation of information, rather than upon information processing skills. In the area of computer-based learning, drill and practice programs have so far tended to dominate. Students and teachers are being tyrannised by curricula which fossilise information into facts to be known, rather than into material to be manipulated and thought about. Even powerful, open-ended tools such as word processors can be used in a fossilised education if they are viewed as neat typewriters rather than tools for text manipulation and information handling which allow us to explore our own thoughts. A major aim of teaching computing in schools should be to help our students to acquire a toolkit of cognitive skills.

The value of the cognitive toolkit is that it frees the mind of its user for more general problem solving activities. Children have little difficulty in acquiring the subskills which form part of the toolkit, although the integration of relevant skills may be another matter. The educator has the choice of which subskills to emphasise, of course, and how to present them. Reciting multiplication tables, or writing row after row of well formed letters of the alphabet presents the child with subskills for his or her toolkit. Many students are not motivated by this form

of learning. Also, it is questionable whether these subskills should receive such strong emphasis. They contribute little if anything at all to the development of flexible problem solving skills. They are part of an educational philosophy which emphasises knowing *what* rather than knowing *how*, i.e. they assist the development of declarative rather than procedural knowledge. Papert (1980) points out that personal computers can be used to simulate environments which provide children with the conditions necessary for the reorganisation of their understanding of phenomena concepts, etc., and he argues that children can acquire powerful problem solving tools very effectively by learning to program. By programming the computer to create graphic displays, abstract ideas can be made concrete, and the means of manipulating the world can be made personal and apparent.

Computer-based activities which can be used to facilitate the development of the toolkit include the creation and interrogation of databases, the use of word processing packages and Logo programming. The feature common to these tools is that they are open-ended. In the sense that pencils, pens and typewriters are open-ended, I would suggest that the most useful pieces of educational software are open-ended tools in that they do not provide right or wrong answers but opportunities for the development and exploration of ideas. The aim of these activities is not an end in itself in most cases, but to provide general skills which can be used in the solution of other problems. Component subskills of problem solving activities are performed without much effort and attention because they are highly practised. These over-practised skills are valuable, and computer programs which facilitate the development of a cognitive toolkit of subskills are to be welcomed, but they must be chosen with care. Practice does not necessitate drill and practice, and subskills can be acquired in the context of problem solving activities such as simulation games and Logo programming.

The claim made for having children learn to program is that, through interaction with the physical microworld, they will acquire a toolkit of general problem solving subskills. Programming is a specific form of problem olving -- e.g. the problem might be one of how to get the computer to draw a spiral, or a row of houses -- and the act of programming itself requires the use of a set of subskills which are independent of any body of facts about the programming language itself. Arguments supporting the view that certain subskills, which might be developed by learning to program, will improve cognitive processing and performance more generally include the following:

- Programming requires creative and critical thinking to be made explicit. There is no magic button when it comes to giving instructions about the movement of the cursor: it moves exactly as instructed and makes no assumptions of its own. Imprecise instructions are not recognised, and reformulation of imprecise commands is necessary.
- Programming provides an environment in which general concepts, such as *variable, function, transformation,* and *recursion*, can be learnt and their consequences demonstrated. These powerful abstract concepts can be seen in operation through programs. Logo programs are particularly valuable because of their graphical output which can be said to make thinking visible.
- With problem solving through programming, it is possible to appreciate the usefulnes of heuristic approaches to a solution. These are the general problem solving skills involved in planning the route to a solution, solving problems by breaking them into smaller parts, and solving problems by analogy. In particular, programming can foster the idea that problem solving can be organised by parts. Small procedures can be seen as the building blocks from which large solutions are derived.
- The interactive process of getting a program to run as intended gives an appreciation of debugging a less than desired solution. Errors become important and helpful aids to learning. They can be informative as a diagnostic means of locating the source of a student's difficulty with a task. The strategy of using errors as a starting point for remediation and improvement can be generalised to other problem solving tasks.
- The acquisition of the vocabulary of programming, made necessary not only as a tool for thinking but also as a tool for communication with others -- i.e. to openly discuss the process of problem solving during programming -- leads to stronger awareness of the processes of problem solving. Reflection becomes a means of selecting and giving strength to the control processes which are necessary in the choices between alternative routes to a solution, and in the reviewing of resources necessary for problem solving. This reflectiveness and awareness of process has been labelled metacognition, i.e. knowledge about one's own personal cognitive processes and others', their limitations and their applicability. Awareness of our ability to handle problems includes the knowledge of the ways in which we know ourselves to be capable of solving tasks, and knowledge of the kinds of activities in which we must engage in order to find a solution. An important part of this awareness of problem solving strategies is the recognition that individual problems call for individual solutions. The selection of the most appropriate solution will depend upon cost/benefit analysis of the alternatives by the individual in a particular context.

Such educational effects of learning to program are the *powerful ideas* to which Papert (1980) refers. They are powerful tools because once they have been acquired they can be used over and over again. They are powerful in the same way as being able to read is a powerful tool, because it allows the individual to read a variety of materials for a variety of purposes and with important consequences. They are also powerful in another sense, because by making the suggested effects so explicit they have been made testable, i.e. we can evaluate them.

The essence of the claim made here is that programming trains people in problem solving. The programmer must be able to formulate goals and routes to

the goals precisely, to be able to apply a number of heuristics which apply to many types of problems, and to appreciate the usefnlness of diagnostic errors. Once these powerful ideas have been acquired through programming experience, they can then be applied to other kinds of problems. They are generalisable tools which can be applied to a variety of situations with and without computers.

Few of our students will need, as adults, to be able to program anything more sophisticated than a washing machine or a video recorder, and there are very few advocates of programming as an educational discipline in its own right. The benefit of learning computer programming lies in the experience it provides of a range of concepts which are a useful addition to our students' cognitive toolkits.

Papert (1980) goes further than this, however, and points to Logo programming experience as a simulation of the kinds of interaction with the world which can foster cognitive development. He suggests that the reorganisation of young minds occurs through the discovery of regularities in the world and through the testing of hypotheses about its structure. By exploring a computer-based microworld, students can discover the regularities of that environment, and will discover the effects of their own actions upon the microworld. These interactions will produce successive reorganisations of the child's understanding of how the world works. Programming could, thus, accelerate a child's normal course of cognitive development.

Chandler (1984) is particularly concerned that children should not only be aware of, and have access to, national and international databases, but that they also need to be contributors to the store of knowledge they contain. He argues that *guided tours* of someone else's frame of reference are not enough. One of the most effective ways of understanding any body of knowledge is to reconstruct it. In some cases this corresponds to the building of a physical model, or of a computer model with, for example, Logo graphics, and in other cases this may be the redescription or adaptation of a body of knowledge with a personal database. Children need to create their own systems for communicating information with one another, otherwise they will become passive, if not alienated, consumers of the knowledge of others. Papert defined this as the *power principle* . The learner must be empowered to perform personally meaningful projects. The premise that the child's cognitive performance will improve over a wide range of measures, if the educational experience builds upon the child's own experiences, is one which finds ready acceptance among practising teachers and theorists alike. Papert (1980) described this as the *continuity principle*.

Major Toolkit Applications

One might focus on three applications of problem solving in which the toolkit provided by computing is of particular relevance. In the first application the use of computers provides a vehicle for direct problem solving. The second application arises when the computer is used as a tool for creating an object, idea, system, etc. The third application involves problem solving processes in order to find new nses of computers by students. The three areas of application are discussed in further detail in this section.

1 Direct Problem Solving Some examples of the use of computers for direct problem solving are the use of Logo, adventure games and using the computer to control and monitor external devices. The identification and nse of strategies in computing can make it clear when students have engaged in different levels of problem solving activity. It can also help students to invent their own problems and adventures. A useful kind of activity for this is Lego-Logo.

Using Lego-Logo, students in the Coombabah project have built, for example, cars which can go backwards and forwards, a barrier that goes up and down, traffic lights which change, a *drink machine* that gives change, etc. The motor for each model was controlled by the computer through relatively simple Logo programming. For example, students worked towards achieving a sequence where the car moved up to the barrier, which would then rise, allow the car through, and then lower again. The students had to work through four major stages: (1) to comprehend the requirements of the task, (2) to identify and use the combinations of switch commands needed to operate the car and barrier, (3) to convert these into flexible subroutines, and (4) to relate these subroutines to analyses of the distance, times, etc., needed to solve the original problem.

The students structured each of these stages using processes which involved activities ranging from looking at their brief user guides, through trying out possible combinations of switches, to deciding on a sequence of ordered experiments. After solving the original problem, the students were able to provide observers with a detailed verbal description of the stages they had worked through. The feelings of one student are reflected in the following personal report:

> I think this project was about how to make things work [move]. It was interesting and fun. It was hard at first, because we had to find what every bit did. Then we had to think about what we were going to do and make a decision. Then we did lots of test runs and calculations. In the end we ended up with the car moving forward, a gate lifting, the car going under and the gate going down. (Personal communication of a Year 6 student)

The use of programming processes and an appreciation of levels or stages are very apparent in this student's reflection on the problem solving to which he contributed.

2 Creating Something When the computer is used to create some object, idea, process, etc., it is important to alert students (and other users) to the *constraints* imposed by hardware, software or both. The use of any type of technology adds its own specific constraints to the realisation of particular outcomes. The properties of the machine and/or software have to be analysed as a part of the creative activitv. The use of problem solving processes for this analysis has led to

interesting developments. In most instances, the students themselves have been able to articulate the way in which the use of the technology affected the outcomes of their activities. Some students have also been able to specify the concepts involved and what they have learnt.

The following example might illustrate this application. A group of students decided to produce a school newspaper by using a simple desktop publishing package. They learnt about the structuring of the production process by means of activities such as gathering information analysing and evaluating it, deciding on what information to use and how to present it. However, these processes also led them to new and important knowledge and understanding about ways of using language. Two insights, in particular, were the direct result of the consideration of constraints related to the technology. One such constraint was lack of space, as the software was suitable only for articles up to the length of 60 words. This forced the students to consider the style of writing which would be suitable. They decided that they should use as few words as possible to convey ideas. One student coined the phrase *punchy style* to describe what she perceived to be required.

Considering the constraints of the technology led to the consideration of other constraints such as the need to sell copies of the finished product.This meant that the text had to be readable and easily understood by students at different year levels. Another constraint was that headlines were compulsory, so these had to be invented and designed to catch the eye of potential readers.

These constraints led to the students having to edit the work of other students who had submitted articles. The idea of students editing and rewriting the work submitted by peers would not have come up if it had not been for the constraints of the situation. Working in a problem solving climate enabled the students to make decisions naturally, and also to articulate how they reached them. Upon task completion, the students were asked to reflect on their experience of producing the newspaper and what they thought they had learnt. Many of the comments related to setting out and other production components of the task, but a number of students reported that they felt that they had learnt to work more systematically, to plan and to think more clearly.

3 New Uses of Computers A third application involves using problem solving processes in order to provide an alternative approach to new uses of computers by students. At present the introduction of computers into classrooms is often highly structured and prescriptive, with an emphasis on *training* rather than *understanding*. Many guides for the use of computers in schools, such as published user guides, provide step-by-step instructions without any explanation and discussion of the principles. The result can be a lack of flexibility for the learner because of a lack of basic understanding.

In a problem solving climate, the use of new technology is seen as a situation to be dealt with by usual methods of finding information, thinking, making decisions, monitoring and evaluation. Students learning to use new technology

expect to gain information both from the teacher and peers, and to a lesser degree from books and user guides. They expect to use trial and error methods for some details, but tend to look for general principles. The natural processes of enquiry and learning are thus perfectly appropriate, and can be expected to relate easily to using the features offered by the new technology in the classroom context.

One aspect of cognitive and personal development which will be of increasing importance in the future is the ability to be flexible in a rapidly changing society. One way of encouraging such flexibility is to place more emphasis on the use of higher order analytical thinking by students. The Coombabah SUNRISE Project and other opportunities for learning with computers are certainly helping to achieve this by fostering the bond between problem solving and computers in the classroom.

STEPS TOWARDS INITIAL MASTERY OF PROGRAMMING

An arbitrarily selected set of steps towards initial mastery of basic aspects of computer programming may actually suggest how links between problem solving in programming and problem solving in other domains may arise.

Such a set of steps for programming instruction might offer guidance for understanding what might constitute appropriate student experiences and also provide a standard against which to measure instructional methods. These steps are not intended to be an exhaustive list of all possible activities, but rather to identify some of the activities involving considerable cognitive skill. The steps describe a direction which instruction might take, but it must be remembered that gains in problem solving ability occur slowly. Introductory instruction can start students off in the right direction, by making explicit the aspects of programming that are likely to generalise to other domains. Table 2.1 summarises and describes some possible steps.

Language Features

In order to be able to use the programming language being studied, it is important to understand many of the language features or non-decomposable elements of the language. In programming courses teachers typically introduce language features, explain how they work, and have students use them. The teachers in the SUNRISE classrooms at Coombabah explain the language features to the group as a whole, then demonstrate how they work to a small group who become *experts*. Rather than giving students formal practice in using the features, they are introduced to them and then encouraged to explore them for themselves. When difficulties arise, the students consult an *expert* and/or the teacher.

Students' knowledge of language features could be assessed by comprehension items which require the student to predict how programs using certain

Table 2.1 Possible Steps towards Initial Mastery in Programming

Step 1: Command of the **language.**

The features of the computer language are the *primitives* or non-decomposable elements of programming. For BASIC they include *IF . . . THEN, GOTO, PRINT,* etc. For LOGO they include *MAKE, SET, REPEAT, PRINT, TO, IF,* etc.

Step 2: Skills to **design** programs.

This requires the acquisition of a repertoire of templates and procedural skills. ***Templates*** are fixed patterns of code (prototypes) using more than a single feature of the language. They are employed in programs to perform commonly encountered tasks. Templates can perform complex activities such as sorting or searching for words or numbers, and can be used in many different situations. For example, rather than inventing a solution each time sorting is required, a *sort* template is applied to each sorting problem. ***Procedural skills*** are used to combine templates or language features to solve a problem. They include *planning* a solution using available templates and language features, *testing* the plan to ascertain whether it accomplishes the objectives and *reformulating* the plan until it succeeds.

Step 3: **Problem solving skills** which are transferable.

What has been learnt could transfer to a new programming language, a database management system, a computer controlled device, or even a subject matter like *motion* in physics. One aim is to develop a repertoire of *generalisable* templates suitable for adaptation to other tasks and contexts. Another aim is to identify generalisable procedural skills for planning, testing and reformulating problems in a variety of situations. This involves identifying the isomorphism between procedural skills used for several types of problems.

features will perform. In addition, language knowledge can be assessed by asking students to reformulate or change a language feature in a program so that the program does something different. For example, students might change the length of a *loop*, an arithmetic expression or content of a *print* statement. Thus, comprehension and reformulation items can be used to assess understanding of language features.

Students need to learn language features to be able to use the language. However, such knowledge is not sufficient for improving problem solving. Many students who have an understanding of the major language features are not able to compose programs which contain groups of commands working in concert. These students can copy programs, they can change *print* messages in programs, they can modify games, databases, etc., and they can in other ways alter single lines of programs. In spite of the limitations inherent in instructions consisting solely of language features, the computer language is often the only topic addressed in computer texts and courses. Such texts and courses of computing emphasise learning of language features rather than how to use them to solve problems. In the SUNRISE classrooms at Coombabah, students are encouraged

from an early stage to write programs (even animated stories) by exploring, creating and combining procedures.

Design Skills

Design skills are the group of techniques used to combine language features to form a program that solves a particular problem. These include templates and procedural skills. Design skills are essential in order for students to write computer programs

Templates Templates are fixed patterns of code that use more than a single feature of the language. Templates perform complex functions such as sorting alphabetically and/or numerically, counting the number of words in a text, executing basic arithmetical functions, etc. Templates can be stored, called up and used each time a given task is encountered. They perform a function similar to *schemata*, and more specifically *weak schemata* in the theoretical formulation of Anderson (1984)[1] and to *plans* in the work of Soloway & Ehrlich (1984).

When students have a repertoire of templates, they have a set of flexible and powerful techniques which allow them to complete many tasks without inventing new code. Well chosen templates facilitate good programming, because they can help to reduce the cognitive demands of programming by providing obvious ways to decompose a task. For example, in Logo and in a number of other languages, students can form a type of looping template that combines a decisions based on the *IF . . . THEN* language feature with the *GOTO* feature. A specified action would be performed as long as the decision is affirmative. Examples of problems which could be solved by using this type of template are as follows:

- read names *if* the last one has not been located,
- read scores which are smaller than 50 (*< 50*),
- add water *if* the pool has less than 10,000 gallons in it.

This type of template offers a good way to manage flow-of-control, i.e. the order in which the computer executes statements in a program. Programs with sequential and organised flow-of-control are easy to understand and revise. Students can decompose tasks into pieces of the size of their available templates.

The theory is that if one has a strong schema, comprehension is principle driven and predictions can be thought of as derived. With a weak schema comprehension is precedent driven. Predictions are not so much derived as looked up, and generalisations are local in scope and treated with caution. Anderson's hypothesis is that the notion of a weak schema gives the best account of the thinking of ordinary people in ordinary circumstances dealing with ordinary matters of knowledge

Then they can complete the task, by solving the problem using their templates. Students who have a repertoire of templates can write more complicated programs than those without such a repertoire.

Experts structure their knowledge of programming in templates (e.g. Atwood & Ramsey, 1978; Kurland, Mawby & Cahir, 1984). As research in cognitive psychology suggests (e.g. Chi, Feltovich & Glaser, 1980), the way knowledge is structured and organised determines how it will be used subsequently. If programmers develop a repertoire of effective templates they are likely to program more effectively, and to identify and re-use algorithms for problem solutions rather than re-invent solution paths over and over again. Students need to recognise the importance of templates and to collect a powerful set of them.

Templates can be learnt from direct instruction. Some computing textbooks emphasise their usefulness as a mode of acquiring and and organising programming knowledge. However, only a small number of the students in the Coombabah project look for books on computing in the library. They do not make use of templates as frequently as they could. Although this group of students may not be typical of other students learning with computers, it must be emphasised that teachers should motivate and encourage students to look not only at one another's programming but at books containing new ideas. Many expert programmers report that they learn new templates by reading programs written by others.

Students can often employ templates which they could not invent themselves. Most of the Coombabah students know this. By using these templates they can profit from the work of expert programmers and themselves solve more interesting and complex tasks than would be possible if they had to invent all their own templates. Even if they do not fully understand a template initially, the experience of using a well designed one will help students comprehend the technique in the template and the role of templates more generally. Just as experts learn templates from others, so can novices, thus increasing their template repertoires.

Procedural Skills Procedural skills are used to combine templates and language features, which are available to the programmer, in order to solve a new problem. Procedural skills include planning the solution path, testing the plan and reformulating the plan if any of the tests fail. Reformulating, previously mentioned as a technique for testing knowledge of language features, can also be used to modify longer sequences of code.

Programmers need a plan for combining language features and templates to solve a programming problem. They decompose the problem into component parts and plan how best to combine those parts. Once a plan is implemented, the programmer needs to test the plan in order to ascertain its correctness. Testing involves determining whether a program meets specifications by deciding what data or other conditions might cause difficulty and then running the program under those conditions to see whether it operates correctly. When the testing of a program reveals problems, the programmer decides whether it requires refinement. Programmers reformulate programs to make them more adequate.

Planning Planning is required for the solving of complex tasks. Novices rarely work on programs complex enough to demand planning. Programming assignments which involve only linear combinations of single language features often fail to illustrate the advantages of planning. Thus, programming instruction must be carefully designed in order to ensure that students understand the importance of planning and have the opportunity to practise it. Only then can they gain knowledge about the conditions under which a template will function. Planning is an important component of the behaviour of expert programmers. In some studies, experts spend much of their time engaged in planning. In contrast, planning is not an aspect of novice behaviour (Dalbey, Tourniaire & Linn, 1986). Similar differences in the time spent planning solution paths are reported for experts and novices solving non-computer problems in physics (e.g. Larkin, McDermott, Simon & Simon, 1980).

Testing Testing is an important component skill of programming that can be enhanced by asking students to find out whether programs perform as expected or intended. Experts and novices differ in this skill. Experts not only recognise the advantages of testing their programs but are good at devising tests to reveal possible problems. For example, experts tend to test the boundary conditions, to ensure that no division by zero is possible, and to consider difficulties resulting from interactions between parts of their programs. In addition, programs written by experts tend to have built-in tests for potential confusions, such as tests to be sure that the input data meet the problem specifications. In contrast, novice programmers in Years 6 and 7 test only the obvious or usual forms of input and may fail to test all of the code.

Reformulating Reformulating is required when students are required to modify a program plan. It is another skill that differentiates experts and novices. Experts are likely to respond to the results of tests by considering large scale as well as minor reformulations of their programs. In contrast, novices tend to seek localised remediation for their programs, perhaps never learning how to revise larger programs. These efforts of novices often result in what experts have called *spaghetti code*.

The acquisition of design skills can be assessed by asking students to write programs to solve tasks. To require planning, testing, and major reformulations, such problems must be cognitively demanding for the students. This means that the tasks must be reasonably complex, challenging and, where appropriate, have multiple solutions. To measure template acquisition, problems must require commonly learnt templates. Once a problem is solved, the program that solved it can become a new template. As noted above, expert programmers use these skills effectively, whereas novices often fail to plan their problem solutions, fail to test their programs and are unable to successfully alter the programs that do not perform the desired task.

Problem Solving Skills

The major potential gain in these steps towards initial mastery of programming consists of problem solving skills which could also be useful for the acquisition of new knowledge and transfer of prior learning to new contexts. These problem solving skills include both *generalisable templates* and *generalisable procedural skills*, i.e. templates and procedures which are common to many formal systems. They are generalisable in the sense that the characteristics applicable from one formal system to another are made explicit and can be applied to new systems. Learners who represent for themselves their programming knowledge and experience in such a way that the elements common to several programming languages are separable from those that are idiosyncratic to one particular language will be more likely to acquire generalisable skills.

For example, templates such as sorting in one programming language can often be used when programming in a new language. Generalising a *sort* template from one programming language to another may require substituting one type of looping for another. The generalised template is represented in such a way that the user knows that the looping structure needs to conform to the conventions of the new language.

Generalised procedural skills consist of techniques of planning, testing and reformulating which can be applied to several tasks and contexts. For example, there are many similarities between planning a solution to a computer program, an algebra word problem, and a geometric proof. Instructions which make these similarities explicit and provide opportunities to use these skills in more than one formal system may well facilitate the acquisition of general problem solving skills. As yet few programming courses offer opportunities to examine this possibility. Studies of experts suggest that transfers of knowledge and skill are achievable. The procedural skills of planning, testing and reformulating are applicable both in learning new programming languages and in learning to use other systems such as database management software, spreadsheets and word processors.

General problem solving skills may be acquired when students attempt to apply templates or procedural skills learnt in one context to a new context (e.g. Dalbey & Linn, 1986). Also, students may identify aspects of templates or procedural skills that are central to their effectiveness as well as aspects that are peripheral to their effectiveness. This knowledge then becomes general enough to be used for problems in other programming languages and for non-programming problems. The acquisition of problem solving skills can be assessed by asking students to solve problems using an unfamiliar system such as a new programming language. The set of steps of cognitive accomplishments that culminate in technological expertise is a long one. Fairly complex problems are required before students can be expected to use cognitively demanding skills such as planning. Experience with several formal systems may be needed before students actually acquire transferable problem solving skills.

MAKING LEARNING WITH COMPUTERS COGNITIVELY DEMANDING

Jerome Bruner (1966) argued that positive attitudes towards learning are encouraged by the complexity and challenge of the task in hand. Complexity encourages curiosity, and perceptual curiosity in turn generates a state of high arousal or excitement which is relieved by the exploration of the stimulus. Bruner's challenge to educators is for them to find ways and means of fostering the drive to achieve competence at a task, and to create in the student the need to have mastery over the environment through at least some understanding of its complexity.

There is a general demand for higher order cognitive outcomes from schooling. Expressions of the desire to transform Australia into a *clever country* call for the *new basics* of the 21st century, i.e. thinking skills that allow individuals to cope with rapid technological, social and scientific advances, and with the accompanying philosophical, economic, cultural and personal changes. What is being emphasised in countless public statements and reports is the need for teaching and learning which will foster problem solving and can prepare students not only to deal with new technologies as they become available, but to make these technologies part of the student's personal repertoire of intellectual tools. Learning with personal computers can help students towards these aims.

It is easy to mistake sophisticated technology for sophisticated learning, and it would be a mistake to assume that productive outcomes will necessarily emerge whenever people use computers. There are many examples of complex technology being used to achieve low level educational goals, and some uses of drill and practice programs are a case in point here. The computer is a versatile piece of equipment which can be used to promote sophisticated learning strategies in which the machine, the student, or both, take a more active part in the learning experience. It is this very versatility which raises fundamental issues about future directions of educational computing. The computer may be used to great effect as a calculator, a teaching machine a processor of complex information, a creator of microworlds or to control complex systems. In essence, this means that the computer can support a full range of educational philosophies, e.g. acting as a tutor for those who believe we should return to a traditional basic skills curriculum or as a key factor in stimulating the dynamic processes of creative writing, complex problem solving and other higher order cognitive activities.

All too often we find students whose main motive for preferring to learn with a computer is the ever-patient and generally non-judgmental response computing appears to present to their mistakes. The practice of discrete skills is not inherently wrong, in fact, practice is vital if skills are to reach the level of automaticity necessary to allow the individual to focus attention on higher level problems. If we must concentrate on the spellings of words and on the formation of letters with a pencil, then we have less time available to think about the

meanings of the sentences being composed. If the lower order skills are automatised, the students' minds become free to plan, create and review. Tedious routine activities then take care of themselves, but only when they have been practised and overpractised. In the first instance, drill and practice serves the purpose of releasing our minds.

Dede (1986) suggested that instructional control strategies for the use of computers in education form a continuum based on the balance between varying levels of passivity of the computer and the child. At one end lies the directed learning in which students are passive recipients of wisdom and unable to explore the material themselves. At the other end lie the open-ended computing tools such as Logo, data bases and word processing. These problem solving tools give control to the learner but no longer provide built-in guidance when the student has difficulty, although they inform the user that an error has been made or an inappropriate action has been taken. One end of this continuum will appeal to those teachers who believe that there is a critical, generally agreed upon body of existing prerequisite knowledge which all students need to be taught. The other extreme will appeal to the teachers who believe in the need for the student to discover his or her own truths, and to build up their personal knowledge with varying degrees of support from the teacher.

Characteristics of Cognitively Demanding Activities

Several features of learning with computers are likely to increase the quantity and quality of cognitively demanding activities offered to students. For example, the *Assessing the Cognitive Consequences of Computer Environments for Learning (ACCCEL)* project (Dalbey, Tourniaire & Linn, 1986) has identified six features of learning with computers which result in the capacity of such environments to provide cognitively demanding activities. Three of these are characteristic of many school environments, the other three somewhat unique to learning with computers.

The first feature common to some traditional and some computer learning environments is *complexity*. Computers can help students solve complex tasks. For example, students can solve problems which require the management of large amounts of information, such as plotting graphs or computing compound interest. The second feature is *challenge*. The computer can challenge the student to solve problems such as figuring out the best move in a game or determining the most efficient path through a maze. The third feature is the provision of *multiple solutions* to a question or problem. Students can write and compare several programs which accomplish the same end. These three features are common to computer learning environments but they are also characteristic of some traditional learning contexts.

Three additional features of the computer environment are less characteristic of other classroom learning: (1) the computer environment is *interactive* at all times. The computer can respond immediately and informatively to the learner's specific request or need. Thus students can try several approaches to reformulating their computer program and determine whether each of the approaches is successful. In contrast, it can take days or weeks for students to receive responses to their homework or class assignments. (2) Computer feedback is *precise*. It reacts to all input. Without computers, students frequently receive rather imprecise feedback, e.g. an A, B- or a C. (3) Computer learning environments are *consistent*. The same response is received for all identical input. Moreover, for identical input the same response is received by all learners. In contrast, teachers do not necessarily respond identically to the same student reactions, either because they are rushed or distracted, or because they are tailoring their responses to the perceived needs of the individual student. When teachers behave as good tutors, their tailored responses to the student provide advantages not available in most computer-based learning. On the other hand, if teachers are distracted, their inconsistent responses can be less than desirable than those characteristic of computers.

Currently the most cognitively demanding activity readily available for students on the computer is programming. This situation is changing as new software becomes available. Eventually, software which demands higher cognitive skills, but is free from some of the drawbacks of programming, may well become preferable to programming for fostering higher cognitive skills. Currently, however, school students who have cognitively demanding interactions with computers are usually engaged in programming. In spite of this situation, much current programming instruction, including that provided in the SUNRISE classrooms at Coombabah, lacks a conceptual framework, and is not necessarily geared to fostering higher cognitive skills (cf. Linn, 1984). Apart from the software manual, teachers in Australia and other Western countries tend not to have textbooks for programming instruction: instead they often amass materials somewhat haphazardly. Very limited funds have been available for the much needed professional development of teachers in educational computing and other areas of computer education, as many schools and regions have followed a *buy hardware now, plan for its use later* approach. Before programming instruction can achieve reasonable goals, this situation must be rectified. A preliminary approach might be to identify some of the characteristic behaviours of expert programmers, and to attempt to develop such behaviours in our novice students.

Characteristics of Programming Experts

The contrast between good professional computer programmers and children learning to program in school suggests a need for materials which can build up knowledge and skills which might culminate in programming expertise. As matters stand now, some staff teaching computing in tertiary institutions

complain that what students have learnt in school actually interferes with the ability of students to profit from tertiary courses in computing. Analyses of the nature of expertise, the characteristics of current instruction in computing in schools and tertiary institutions, and the potential of the environment in which programming instruction takes place, need to be conducted before the usefulness and relevance of school versus university and college computing can be established.

The behaviour of programming experts contrasts sharply with the characteristics of current instruction in educational computing, as revealed by several studies of experts (e.g. Pea & Kurland, 1984; Linn, 1984; Jeffries, Turner, Polson & Atwood, 1981). To develop the knowledge and skills which might be prerequisites to programming expertise, students must learn the skills experts use every day. Current instruction may not provide this opportunity.

One component of expertise is an extensive repertoire of *programming templates* or procedures. Templates can apply to a whole program as exemplified in an *input-process-output* template. As noted previously, templates can also apply to a specific function of a program. Research in the USA and at Coombabah has shown that both expert adult programmers and quite young students who are *experts* in some aspects of programming can articulate their templates, recognise the relationships between their templates and new templates, and actively seek to create new templates or procedures.

Professional programmers use a variety of *procedural skills*. As noted above, these are among the skills referred to as the *new basics* of schooling in many recent reports. They are part of the set of thinking or problem solving skills which individuals need to survive in our society. Important components of these skills are planning and the ability to determine an appropriate sequence of available procedures. In the past, investigations of expert performance in formal systems, such as solving mechanics problems in physics, have proved themselves to be informative for educators designing programs to foster these skills in novices and to make the learning of programming more cognitively demanding (e.g. diSessa, 1982, 1986, 1988a, 1988b; Larkin, McDermott, Simon & Simon, 1980).

The literature dealing with the planning of solutions to programming problems by experts indicates that experts engage in two complementary techniques: top-down design and stepwise refinement (e.g. Brooks, 1980; Atwood & Ramsey, 1978; Jeffries, Turner, Polson & Atwood, 1981). *Top-down design* is an approach which decomposes a complex problem into subproblems. Experts can do this effectively, we surmise, because they have a large repertoire of program templates. Experts use their knowledge of templates to guide the decomposition process. Top-down design is somewhat iterative in nature. After the initial decomposition, each resulting subproblem may require further decomposition until the task reaches a manageable degree of complexity. Experts proceed with top-down design by selecting appropriate templates for each problem.

Stepwise refinement experts engage in successive restatements of the problem specification, with each step coming closer to machine level notation. The original problem specification describes, in natural language, a process the computer is to perform. Stepwise refinement means to translate the process description, through incremental stages, into language, i.e. code, which the machine understands. Experts can do this well because they are very familiar with the language the machine uses. Experts know the degree of precision and the degree of clarity needed to describe the process for a machine solution. Ultimately they generate unambiguous statements of their program design.

Characteristics of Students of Programming

Students who are just beginning to learn a programming language usually differ dramatically from one another. There are many reasons for this which relate to differences in motivation and interest, previous experience, ability and other variables contributing to individual differences between students. Another important source of variation in learning outcomes relates to the kind of instruction which the students have received (e.g. Soloway, Ehrlich Bonar & Greenspan, 1982). For example, Dalbey, Tourniaire & Linn (1986) observed 30 junior high school BASIC programming classes. Most were found to have offered teaching which emphasised features of the programming language, and often failed to provide instruction in how to combine the language features into larger algorithms.

In this and other research it was found that students are introduced to a language feature such as the *PRINT* statement, and then write programs using that statement. Their understanding of the program is basically at the level of a single line. They type in a line and get feedback about their use of the *PRINT* statement. Students respond by typing in a different line which hopefully corrects the mistake they have made initially. These students are engaged in drill and practice on a language feature. Instruction rarely emphasises the templates which experts use for solving programming problems. Students therefore fail to acquire procedures which help them decompose problems and plan problem solutions.

Novice programmers are characterised by a *rush to the computer*. They frequently attempt to go from a statement of the problem directly to trial and error of program code without any consideration of how to design the code. Novices appear to lack the tools necessary for constructing intermediate states between the problem specification and the problem program code. They rarely receive an opportunity to observe their teachers or expert programmers model the use of planning.

The expert process of *stepwise refinement* appears not to be required or really necessary for most assignments novices receive in programming instruction. It appears that many students fail to grasp the notion that programs are detailed process descriptions which can be refined out of natural language description.

They presume that programs are assembled by piecing the language features together. They fail to understand that the natural language problem description is less precise and more ambiguous than a problem description in machine terminology, i.e. code. They do not engage in the activities required for refining the natural language statement of the problem into a statement which can be decomposed and coded into a problem solution. As a result, when they are asked to solve problems which are more complex than simple translations of known language features, their solutions are often poorly organised, inefficient or even incorrect. The top-down designs and stepwise refinement which experts use to write programs are not taught or modelled by the teacher. As discussed in Chapters 6 and 7, a conservative estimate is that 25% to 35% of the students in the SUNRISE classrooms at Coombabah appear to be lacking the tools necessary for constructing intermediate states between the problem specification and the program code.

Explicit Intervention to Foster Higher Cognitive Skills

One method of intervention to counteract the above described lack of programming sophistication in novices is to provide students with some abstract templates which they could use for stepwise refinement of the problem specifications. This would equip students with a mechanism for constructing a problem solution that might be more detailed than the available problem specification but less detailed than the actual language statements. Students would thus be encouraged to consider an intermediate state between the problem specifications and the program code.

As noted above, most students move directly from the problem specification to the keyboard. This could lead to frustration and inefficient trial and error solutions. Coombabah students are being given tasks that can usually be solved or partially solved. In some areas they ask each other and keep trying. Lack of planning and inefficient solutions were certainly evident. The majority of students did not appear to become frustrated initially. However, as will be discussed further in later chapters, there was an indication of increasing frustration with more experience, with some Year 7 students becoming quite alienated.

It should be noted that students basically appear to be very happy when working at the terminal or on their laptops. They fail to associate their difficulties in achieving a solution with their lack of planning. A general belief among students is that more computing time is needed, rather than that they need to plan, hypothesise, evaluate, etc., to solve problems effectively.

Because of the nature of most of the instruction in computing, it is impossible to emphasise planning until the students have acquired at least a reasonable subset of the features of the computer language being taught. At Coombabah, the teachers have tried to emphasise planning but they have not explicitly taught it.

They did not provide abstract procedures which students could use to practise refining the problem specifications, nor did they provide students with exercises which required the translation of problem specifications into the provided procedures or templates. Students need practice in coding their solutions from templates as well as in mapping problem specifications onto templates.

The literature in educational computing shows that current modes of instruction frequently fail to communicate the value of planning in programming. This lack of appreciation of planning stems in part from certain characteristics of instruction. (1) Students' initial programming experiences do not require planning, therefore the advantages of planning are not apparent to them. (2) Students find the computing, i.e. interaction with the computer, very motivating and they prefer to be interacting with the computer , even if they are not making progress in solving the problems. Unfortunately, the interactive nature of the computer learning environment is not being well channelled towards the development of the higher cognitive skills such as planning.

One important reason why students fail to appreciate planning is that many of them can solve even the most difficult problems assigned without planning. Many students could recognise how to solve the problems which were presented to them without spending any time in planning.

Our experience at Coombabah suggests two directions which teachers might wish to consider in their endeavour to make instruction more cognitively demanding for their students:

Rather than beginning programming instruction with drill and practice in the language features it would seem quite appropriate to begin instruction with comprehension of program code. Students could be given reasonably sized programs (10-15 lines of code) and could be encouraged to come to understand those programs. Those programs would demonstrate how planning is used in programming. Students could see how experts use planning to write a big program. Thus students would have a better understanding of the role of planning in programming.

2 Structure diagrams could be used to help students comprehend a larger or more complex program. Such a program could be represented using structure diagrams. Comprehension of the program could be facilitated by using structure diagrams to illustrate the templates or procedures used by the programmer to construct the program. Instruction could then proceed by demonstrating the top-down design and the stepwise refinements as used by the expert programmer in the construction of a program.

Programming instruction has the potential of fostering the higher cognitive skills asked for by many recent reports on the state of educational practice but, so far, their potential is not being achieved. Instruction which builds sequences of computing knowledge and skills culminating in the planning skills used by expert programmers requires early and consistent emphasis on these skills. Teachers are

needed who can demonstrate and model good programming. Texts are needed which delineate the steps between problem specification and program code. Research is needed to understand more clearly the sequences of activities which will facilitate the desired learning outcomes.

Metacognitive Outcomes

The impact of working with computers on the problem solving skills of students may be increased if students had a greater awareness of the problem solving procedures or strategies embedded in their work. Brown, Campione & Day (1981) distinguish three types of training programs implemented by educators:

a *Blind training* in which the learner is induced to use a strategy without concurrently understanding its significance.
b *Informed training* in which the learner is persuaded to use a strategy and at the same time provided with some information as to the significance of the activity.
c *Self-controlled training* in which the learners are not only persuaded to use a strategy, but are also explicitly instructed how to employ, monitor, and evaluate the strategy.

Of these three training methods, the self-controlled method is by far the most successful in terms of enhanced performance and transfer. It has long been known that problem solving skills are likely to transfer to novel situations only if the principles on which they are based are made explicit to the learner (e.g. Lochhead, 1985; Simon, 1980). The self-controlled method may thus also be more effective in learning with computers.

One way to persuade students to monitor and evaluate their strategies while working with or without computers is to encourage them to reflect on their actions, for example by *thinking aloud* and by verbalising their ideas and strategies. Verbalisation helps learners to externalise ideas and strategies, to reflect on them and to elaborate them. Students could work in pairs or small groups on a task and take turns in verbalising while the others listen and ask questions. This forces both problem solvers and listeners to evaluate the strategies used and to monitor plans and solutions closely.

Computer-based learning environments are particularly conducive to enhancing cognitive and metacognitive skills as they provide the learners with many opportunities to practise these skills and receive immediate feedback.

CONCLUSION

It is possible to list some benefits which can be associated with learning with computers, in some circumstances. None are guaranteed to automatically flow from computer use, but many can be achieved through good teaching and the modelling of effective learning with and without the computer. There is an expectation that there will be: student access to learning resources and sources of information which do not depend upon the teacher; an increase in the variety of styles of teaching and learning; greater metacognitive awareness in students; more student planning and implementing of their own work programs· greater ability among students to use computers; and an improvement of skills such as self-monitoring, generalising, theory building and verifying information, concepts and ideas.

A number of commentators have offered the view (cf. Ridgway & Passey, 1991) that the experiences which students are offered in schools often conform to the school culture, and not to the intellectual tradition the educators purport to be teaching about. So students can learn how to perform a set of mathematical procedures, but not to function like a mathematician (e.g. Schoenfeld, 1985), or learn scientific facts, but not how to function like scientists (e.g. Edwards & Mercer, 1987). Computers in classrooms can actually offer the opportunity for students to learn how to be mathematicians and scientists. The students will also gain a deeper appreciation and understanding about the products of work in those disciplines.

The real implications of computing in education are yet to evolve. Harry Simon, in drawing an analogy between the introduction of the computer and the invention of the steam engine and the motor car (Simon, 1987), points out that both the steam engine and the motor car were responsible for further technological and scientific developments, but above all they opened up new social worlds, in which people had to function, and for which they were required to develop appropriate skills. Many of the social changes now associated with the motor car, even mediated by it were impossible to predict at the time of its invention: for example, the suburban sprawl, environmental damage and huge employment opportunities.

Major educational and social changes can be expected from the introduction of computers, and these changes will be even larger as a result of the use of computers in learning and teaching. We have to live with the fact that we do not know what the real and best uses of computers in education will be, and adopt a style of research and development which allows us to capitalise upon its as yet undetermined potential in an opportunistic fashion.

Because the cognitive technologies which we invent serve as instruments of cultural redefinition (shaping who we are by changing, and not just expanding and supplementing what we do), defining educational values becomes a foreground issue. The demands of an information society make an explicit emphasis on general cognitive skills a priority. The urgency of updating

education's goals and methods can only be met by an activist research paradigm which simultaneously creates and studies changes in processes and outcomes of human learning with new cognitive and technological tools.

PART II

ACQUIRING COMPUTER LITERACY

3

Curriculum Objectives

This chapter addresses the general question of what we expect to gain from the introduction of educational computing into the curriculum. In considering alternative instructional uses of this technology we will discuss both current practice and future possibilities, and reflect on how educational philosophies and our perceptions of the human/computer relationship can affect the uses made of hardware and software.

As noted previously, the effects of introducing computers into the classroom are neither predictable nor controllable. The tools which manifest themselves in computing are extremely versatile and can support many educational philosophies and objectives. Educators must reflect actively upon which form of education they are aiming for for their students. What is certain, however, is that the computer literacy requirements for the average person will expand dramatically during the 1990s. The least we should aim for might be to somehow provide everyone with the minimal amount of computer knowledge that would enable them to become *computer comfortable*. In this context the meaning of 'computer comfortable' is to allow people to be able to *interact easily and without fear with a computer* at a level appropriate to individual needs.

EDUCATIONAL GOALS

Computer software can be described along a continuum of open-endedness. At one end are the single purpose programs which fulfil only one kind of demand, e.g. worksheets, drill and practice programs and many simulations. At the other end are the open-ended tools such as word processors and programming languages, with which any number of different outcomes can be achieved. Another way of describing this continuum is in terms of the purposes of computing, i.e. for simple *training and practice* or for use as a *tool*. The training/practice software aims at a known goal or a well defined product, whereas the open-ended computing tools are more like pencils, paint brushes, hammers, chisels, etc., in that they are not designed for the achievement of a particular product or goal. Rather, they are tools with which many different and personal goals can be accomplished, and which can be adapted for many purposes. Training/practice software leads to clearly identified *correct* uses and

answers, which reflect the acquired knowledge and skills of the user. At the other end of the continuum correctness depends upon what use is made of the tool and on the capabilities of the user.

Different educational goals are inherent in the use of these different types of software. Teachers who are content to use computers as *teaching* machines, i.e. as providers of automated practice and testing, tend to be identifying themselves with the content-oriented curriculum, and with the view that the aim of learning and teaching is primarily a matter of students' acquiring facts. This approach will lead to the selection of programs which will help students know the facts that teachers and society consider worth knowing. There is strong support for this view in certain influential educational circles, but there are dangers in concentrating only on *what?* at the expense of *how?*, i.e. knowing what to do with the facts one has acquired.

There is little merit in educating children to become walking encyclopaedias. Rather, our society needs problem solvers who have access to both the information relevant to a problem and the strategies for solving it. Computers can remember (i.e. store and access) facts better than people can, so why not rely upon computerised databases as sources of information and use people's energies for problem solving? More than any other educational innovation, the personal computer is useful for both these purposes. It can store and assimilate, in different ways, vast numbers of facts and rules and it can assist in the development of flexibility of thought. Obviously, to make use of a tool one has to know what to do with it as well as how to do it. The ideal philosophy for learning and teaching with computers is probably somewhere between the two extremes. The computer is capable of stimulating and supporting a great variety of educational goals. What is important is that, before educational goals are established for particular uses of computers, and before software is selected, educators clarify for themselves their own educational philosophies. The choices educators make about the use of computers in their classrooms may have profound effects, not only on the cognitive development of children, but also on the nature of education itself.

GETTING STARTED WITH COMPUTING POLICY

Every school and every classroom needs a written statement, no matter how short or tentative, of what role it intends computing to play in its curriculum. Such a statement should probably be part of the school's long-range plan, but may differ from school to school and within a school, and it will evolve with time. By addressing the following questions, a school can begin to come to terms with how it will view the role of computing:

Is computer literacy important enough to have a place in the curriculum?
Is computer literacy necessary for all students in this school, for selected ability groups or for students in certain subject areas?
Which computer skills must students exhibit in order to be regarded as being able to use the computer as a tool?
Should the ability to write computer programs and to debug computer programs be considered a component of computer literacy?
If the answer to question 4 is yes, which computer language should be used?
Which computer related social and political issues should students be able to analyse and evaluate?
Should computer training for computer literacy constitute a separate area of learning, or should it be integrated into existing subject domains?
What knowledge do incoming students have of computers? How can the school's curricula take differences between students into consideration?
At what stage/level should students start to acquire skills in computing? If these skills are to be used as tools to support learning in other areas, they will need to be acquired early.
When and how will the plan to make computing part of the curriculum be implemented?
How will feedback be secured and evaluated?
When and how will learning and teaching of computing be re-evaluated on the basis of student, teacher and institutional experience with curriculum reforms and in relation to the evolution of new software?

computing policy may be written separately or be written into each document in the different curriculum areas. Whichever option is chosen, the major concern must be that educational computing is perceived as something to assist reaching the goals set out in the school's total curriculum policy. What do we expect to gain from using computers in the school? Do we want students to learn *about* computers or *with* computers, or both? To some degree the answer to this question depends on how we define computer literacy.

COMPUTER LITERACY

Much depends on the definition of *computer literacy*. When an Education Ministry has decided to introduce computers into primary as well as secondary schools, it faces large bills for the hardware, software and other materials. In addition it faces the costs of curriculum development, of professional development and support for teachers, and of the assessment of student achievement. It is obvious that the way computer literacy is defined not only determines the contents of curricula and assessment, but also will have a profound effect on the tax-payer's willingness to support the use of computers in school.

Computer Literacy and Other Literacies

Concern has been expressed as to whether the focus on computer literacy might divert attention from or even conceal the need to concentrate on basic reading, writing and number skills. This concern is likely to be unfounded if computer literacy is viewed as an emerging component of the complex concept *literacy* as used more generally. There should be no incompatibility, because, as it develops, computer literacy will provide additional support and reinforce the need for the more traditional types of literacy. Indeed, in an information or knowledge society the ability to read, write and calculate will become more and not less important. The introduction of computer competencies into a curriculum cannot supplant the need for the more traditional literacies.

A report commissioned by the Club of Rome (Botkin, Elmandjra & Malitza, 1979) called for global educational reform to foster what it called innovative or *anticipatory* learning. Such learning is designed to prepare individuals to be able to consider possible contingencies and the long-range implications of various choices. Among learning activities particularly conducive to anticipatory learning are forecasting, simulations and modelling. Their use is made easy and inexpensive by low-cost personal computers. Such techniques emphasise the future orientation of innovative learning and the need for individuals to develop the ability to reflect on the implications of different decisions and to evaluate alternative futures. *Anticipatory* learning skills should probably become part of the basic skills of our students in the 1990s and beyond. These considerations should strengthen our efforts to bring about curriculum reforms which address computer literacy. They also help link the various dimensions of computer literacy to each other and to traditional literacy skills.

There is an established field of research which compares preliterate and literate school children to assess the postulate that literacy is a prerequisite of logical reasoning. It has been argued (e.g. Olson, 1978) that literacy allows individuals to master the logical functions of language and to separate them from the interpersonal functions of language. Language literacy assists cultures towards the development of formal reasoning systems. Will computer literacy manifest itself in a similar way?

Olson (1988) has argued that computers play three essential roles in students' learning:

> they can provide rich databases which can be used by individuals as sources of information and for the construction of personal knowledge;
> they permit individuals to organise knowledge in a new way; and
> through interaction with peers about goals, successes and problems in specific computing situations, computing allows for a greater understanding of one's own information processing and problem solving processes, and those of others. It develops metacognitive knowledge and skills.

Computer literacy is of practical importance for all members of our community, young and old alike. For example, all adults need to know enough about computer systems so as not to be intimidated by an error on a bill resulting from a computerised accounting system. Individuals need sufficient knowledge of computers to be able to decide whether to acquire a machine for home or work, and they need to learn how to evaluate when computer applications are helpful and when not. Some of this knowledge can be acquired as a by-product of programming and other computer uses, but most of this type of useful information will not be learnt in this way. Indeed, one might argue that most of what the ordinary citizen needs to know about computers might not be learnt from hands-on computing experience.

Defining Computer Literacy

ERDU's NEWTECH/HITECH/INFTECH Glossary (Queensland Institute of Technology, 1985), one of the rare dictionaries, glossaries or encyclopaedias containing the concept of *computer literacy*, provides the following comment on the term:

> An ill-defined term concerned with computer familiarity. Problems arise in consideration of exactly what skills and knowledge should be possessed by people who, even though they may not actually work with computers, will be living their daily lives in a society dominated by the use of electronic data processing. Opponents of the concept of computer literacy draw analogies with, for example, the automobile: they argue that one may derive all the normal benefits of personal transportation without any knowledge at all of what is going on under the hood/bonnet. However, it is not unreasonable to suggest that anyone who draws such simplistic analogies demonstrates convincingly the need for computer literacy. (p. 16)

It is not surprising that the term *computer literacy* shares the semantic ambiguity of language literacy. The meaning of the latter term is restricted by some to reflect the acquisition of simple reading and writing skills (a *narrow* definition); others understand literacy to be far more than the acquisition of such basic skills.

> Literacy is not the simple ability to read and write: but by possessing and performing these skills we exercise socially approved and approvable talents; in other words, literacy is a socially constructed phenomenon. (Cook-Gumperz, 1986, p. 1)

Researchers at the US Literacy Institute have suggested that there are a multitude of literacies, each of which is an 'integration of ways of thinking, talking, interacting and valuing, in addition to reading and writing' (Education Development Center, 1988, p. 4). Each literacy is embedded in particular social settings, is shaped by children's early experiences in the home and community environments, and is modified by different literacies encountered daily in and out of school. In short, in order for children to be successful at school and in society, they need to master a broad range of literacy competences (almost in the sense of

being multilingual) in order to cope with the diversity they can expect to encounter in written and oral communications across a wide array of contents and contexts.

Widely accepted definitions of computer literacy can be classified into *comprehensive* and *narrow* definitions. Comprehensive definitions describe literacy in terms of the compendium of knowledge and skills which ordinary, educated people need to have in a particular domain (e.g. language, numeracy, health, economics, etc.) in order to function effectively at work and in their private lives in their culture or society for the remainder of this century. Obviously, a literate person can make use of a wider range of intellectual strategies than someone who is not literate. The following examples are of the comprehensive type, defining computer literacy as:

> whatever understanding, skills and attitudes one needs to function effectively within a given social role that directly or indirectly involves computers. (Husen & Postlethwaite, 1985, p. 937)
>
> whatever a person needs to be able to do with computers and know about computers in order to function in an information-based society. (Hunter, 1983, p. 9)
>
> that compendium of knowledge and skills which ordinary educated people need to have about computers in order to function effectively at work and in their private lives. (Haigh, 1985, p. 161)

Comprehensive views of computer literacy go beyond the narrow definition relating to a body of basically technical information and include knowledge of how computers work, how they are used, and their impact on society. Luehrman (1980), for example, believes that this body of subject matter should more appropriately be termed *computer awareness*. He regards computer literacy as a cultural phenomenon which includes the full range of skills, knowledge, understanding, values and relationships necessary to function effectively and comfortable as a member of a computer-based society. He stresses that the computer literacy needs of any one individual will thus vary according to that person's particular involvement with computers. No single approach to computer literacy can serve all audiences and contexts.

To make this comprehensive notion of computer literacy more operational, Watt (1982) divided it into four distinct but interrelated components:

1 *The ability to control and program a computer to achieve a variety of personal, academic and professional goals:* This includes the ability to write programs in one or more computer languages, read, understand, and modify more complex computer programs, to use a computer as a problem solving tool, and to analyse information generated by a computer program (for example, predictions about economic trends or other futures).
2 *The ability to use a variety of pre-programmed computer applications in personal, academic and professional contexts* : This includes the ability to make informal judgments about the suitability of a particular software tool for a particular purpose, and to understand the assumptions, values, and limitations inherent in a particular piece of software.
3 *The ability to make use of ideas from the cultures surrounding computer programming and computer applications as part of an individual's collection of strategies for information retrieval, communication and problem solving:* This aspect of computer literacy corresponds to the effect on intellectual functioning of learning to read and write, and is probably the most difficult to incorporate specifically into educational programs since the effects themselves are not yet well understood. However, the fact that these concepts might be difficult to integrate into school programs at present does not make them any less important to the functioning of a computer literate person.
4 *The ability to understand the growing economic, social, and psychological impact of computers on individuals, groups and society as a whole:* This includes the recognition that computer applications embody particular social values and have different kinds of impacts on individuals and different segments of society. It includes the understanding necessary to play a serious role in the political process by which large and small scale decisions about computer use are made, and to transcend the dependent roles of consumer or victim (cf. e.g. Turkle, 1984; Weizenbaum, 1976; Wessel, 1974).

Some computer literacy objectives can be integrated into mathematics, social studies, English, science, or other curricula. Others do not fit readily, for example the fundamentals of a computer language and computer programming. Computer literacy objectives which do not conveniently fit into existing disciplines can, however, be taught through specially produced modules of basic computing skills.

It would seem that a comprehensive definition of computer literacy addresses at least three sets of issues:

1 using the computer as a tool;
2 determining the need for and, where appropriate, acquiring programming skills; and
3 assessing the personal and societal implications of pervasive computer use.

As individuals and educational institutions grapple with the implications of computer literacy for the curriculum, each of these three issues requires attention. The weighting of them and details of curriculum contents will depend on the aims of specific courses, and whether computer issues and skills are introduced into a curriculum through separate courses or through modifying existing courses in traditional subject domains so that computing is totally integrated in the subject and becomes a major tool for learning.

The comprehensive view is that computer literacy is an understanding of computers and computing that enables one to evaluate computer applications as

well as to *do* computing if this is a personal need. This view of computer literacy fits with the long-established tradition of scientific literacy and related formulations such as technological literacy, geographic literacy, and economic literacy, to name only a few. Scientific literacy is generally defined as the knowledge about science which the lay person needs to function effectively. Scientific literacy refers not only to learning scientific facts but also to one's understanding of the implications of science and science/society issues. Thus, it is not surprising that we often see computer literacy equated with *computers in society* and courses on the social role of computers.

The comprehensive view of computer literacy is also consistent with the recommendations of the US National Council of Teachers of Mathematics (1980). In *An Agenda for Action* they recommend that a computer literacy course, familiarising the student with the role and impact of the computer, should be a part of the general education of every student.

The *narrow* view is that computer literacy is simply a matter of using the computer for particular purposes, i.e. *doing* computing. The advocates of this view tend to define computer literacy as simply *doing computing*. They argue that the most basic components of literacy in a language are that the literate person has the ability to read and write, that is to *do* something with the language, not merely to *recognise* that language is composed of words, to *identify* a letter of the alphabet, or to be *aware* of the pervasive role of language in society. Literacy in mathematics means the ability to add numbers, solve equations, etc., i.e. to *do* mathematics, is not merely to recognise that numbers are written as sets of digits, to recognise a formula or to be aware of the advantages of being able to do mathematics.

By analogy, computer literacy must also mean the ability to *do* computing, and not merely to recognise, identify or be aware of alleged facts about computers and computing. Luehrman (1981) refers to the latter facts as *hearsay knowledge*. This category of knowledge, i.e. hearsay knowledge, the lowest in Plato's hierarchy, is essentially verbal. Its acquisition involves the student mainly in encoding information and remembering it when an appropriate stimulus is presented. It is qualitatively different from the knowledge that comes from experience, i.e. *doing* writing, mathematics, or computing.

The basic flaw in attempting to apply some of the objectives noted in the literature as a standard for what should be taught to achieve computer literacy is that, of the objectives provided, very few actually require the student to be able to *do* anything. For example, in 1979 the Minnesota Educational Computing Consortium (MECC) was awarded a large grant from the US National Science Foundation and came up with 63 objectives. Only 12 of these 63 objectives require the student to actually compute. Eight of the nine objectives in the *Programming and Algorithms* section fall into this category, along with three of the thirteen in *Software and Data Processing*. The other 51 objectives involve nothing more than student acceptance of generally held views or *hearsay knowledge* about computing, such as might be acquired from a book (cf. Johnson, Anderson, Hansen & Klassen, 1980).

Examples of items testing such generally held views about computing in the MECC (1979) list are:

> Identify the five major components of a computer.
> [Correct answer: input equipment, memory unit, control unit, arithmetic unit, output equipment.]
> In addition to input and output equipment computers contain -
> [Correct answer: memory units, control units, arithmetic units].

Clearly, the student who has read and memorised the classical definition of a computer will score full marks on this item. Yet the problem is that months and years can pass in the life of a computer user or even a professional computer programmer without any need to remember, make use of or even reflect upon the fact that somewhere inside the machine lies an arithmetic unit and a few microns away on the same chip lies a control unit, and that they are, logically at least, distinct. A programmer could create an entire management information system or a data analysis package without ever calling on that piece of knowledge or putting it to use. Except for a few who work at or near the hardware level, people who *do* computing rarely use such general knowledge about computers.

An even greater concern about the use of objectives like those noted above comes out of direct experience of working with children and adults who are gaining hands-on experience in computing. After only a few hours of such experience, they know enough to score near the top on the handful of test items based on the dozen MECC objectives requiring that the student be able to *do* computing. Yet at the same time these students tend not to have any idea about what arithmetic units or control units are, and what the difference between them is. However, they do know about *input*, *output* and *processing* because they experience these things and have a need for words to communicate about them. To clarify the difference between this general knowledge and knowledge that comes out of practice, consider the different flavour of the following objectives (which do not relate to general knowledge):

Follow and give correct output for a simple algorithm.
Modify a simple algorithm to accommodate a new, but related task.
Develop an algorithm for solving a specific problem.
Design an elementary data structure for a given application.

Negative Aspects of the Narrow View

Narrow definitions of computer literacy suggest that hands-on computer experience and computer programming are the only important components of computer literacy. Those who promote this philosophy may unwittingly promote

mindless or meaningless *doing* in addition to constructive experiences. Without adequate direction, students who are *doing computing* may actually be practising non-rigorous procedural thinking, acquiring misconceptions, and otherwise wasting valuable learning opportunities. The typical *doing computing* approach consists of teaching students to write a few programs, usually in BASIC. As Papert (1980) points out, this strategy often results in student learning that stifles creativity and suppresses motivation. It can also lead to awkward and poor algorithmic thinking. Hands-on computer experience does not guarantee computer literacy.

Obviously, those who argue for the narrow view neglect the semantic ancestry of the concept of literacy and preclude a broader understanding of computer technology. There are very good reasons for people to both communicate with computers and be knowledgeable about them. The solution lies in teaching computing *not* for its own sake but in providing students with constructive experiences in the use of computers to meet the requirements in all subject areas and in their personal lives.

In the brief time which is usually allocated to the running of a typical computer course, the student can learn little about problem solving and algorithms. It is also difficult to provide experiences with a variety of languages, and to build bridges that will allow for the adaptation of what has been learnt to other and real world domains. Students may end up with limited and inefficient strategies for problem solving, and if students only learn about a single computer and software system, they are in danger of developing a narrow conception of computing and have difficulty in transferring what they have learnt even to other computer environments.

INSTRUCTIONAL OBJECTIVES

The introduction of personal computers into schools has been justified mainly by two arguments. The first argument relates to the need for societal and vocational relevance of education. Computers play an increasingly important part in our everyday lives, and our children should be educated in their use and in the principles of their operation in preparation for their encounters with them in the workplace and elsewhere. Those who support this view tend to believe that the subject *computing* should be introduced into schools.

Although the parental lobby strongly supports it, this view of educational computing is limited. It regards hands-on keyboard experience as all-important, stresses the amplification aspects of computers, and also educational uses for drill and practice, but largely disregards the opportunities computers provide for the development of the computer user's improved power of thinking, problem solving and learning. These issues relate to the second argument.

The second argument revolves around the teaching of programming. The assertion is that experience in computer programming will result in new ways of

thinking, e.g. that the student will improve his/her power of formal reasoning, will solve problems through heuristics and being placed in a position that encourages recognition of flaws within any suggested solutions will become more reflective and a better decision maker. This is a similar argument to that which has been put forward for the relevance of Latin by educationists in the past. Some argue that programming is the new Latin, but both programming and Latin will only be of use outside their own restricted domains if there is a transfer of skills to other domains of learning.

A review of the literature and discussions with teachers provided some information on how teachers view and use computers. Two distinct instructional uses of computers, which reflect the above arguments for the introduction of computing into schools, became evident: computing as a new subject domain and computing as an instructional tool for use in existing subjects. Obvious differences will result from these uses with respect to instructional goals, learning potential, demands on the teacher, pedagogy and curriculum implications. Contextual factors influencing classroom practice and teachers' personal theories of computers and computing are the major determinants of preferences for how computing should be integrated into the curriculum. The novelty and uncertain status of computing in schools may make it problematic for teachers, particularly those who are not volunteers for the teaching with computers, to decide which position to take.

Both positions will be presented in the next section. Many aspects of the curriculum are similar. Their stated educational goals appear not to differ. Both aim to achieve computer literacy in students, though their respective definitions of computer literacy and criteria for its assessment differ.

Doing Computing

Although there are no firm departmental or regional curriculum guidelines for the use of computers in primary schools, junior or middle secondary schools, there is a remarkable consensus among teachers and educational administrators as to what *doing computing* as a subject means. Like any other school subject it has its own terminology, its own underlying theories, sequence and scope of concepts. The goal is computer awareness or literacy, terms which are sometimes but not always synonymous.

The rationale is that students need to be prepared for the society of the future, i.e. the technological age. The theory of those advocating teaching computing as a subject in schools is that exposure to the machine, software and peripherals increases comfort levels of students until a certain proficiency level (commonly a student's ability to use computer software without the teacher's assistance) is reached. The usual method is hands-on experience.

For the classroom teacher there is an inherent problem with the notion of computing as a subject: it is new and often there is no place for it in the

timetable. This means that computing has to be run as a concurrent activity with other, more traditional school subjects. In fact, the teacher has to teach two things at once. Doing several things at once may not be new (teachers frequently run multiple activity groups), but lack of familiarity with computers and the fact that there is usually little relationship between what different students or groups of students are doing on their computers, and/or the rest of the class, raises questions of how this can best be managed.

Given the very loose definition of computer awareness and its accompanying rationale, any software that is crash-proof and user-friendly is suitable. This can include utility programs, drill and practice, tutorials, adventure games, word processing, etc. In most classrooms the only criterion is that the software should require minimum support from the teacher who has to give so much individual attention in a class where students learn with computers. This makes programming, including Logo, problematic (cf. Chapter 4). Although programming is often considered to be one of the advanced stages of awareness, it is not self-sustaining. It demands time and attention, and both are at a premium for the teacher in the classroom.

Strategies for teaching students to use computers under these circumstances usually include an initial teacher demonstration of aspects of the software on the computer, allowing students to teach themselves through trial and error, using supporting documentation, and peer tutoring using the few already computer literate students who are found in most classes.

It is time-consuming for teachers to locate and appraise suitable software and other computer-related curriculum materials, to keep up with the field and provide the outside-lessons tutoring and remediation required by some students. Keeping track of how well students are managing on the computer may also be difficult. Ways of monitoring and evaluating the development of students' computing skills are discussed in later chapters. The exact nature of computer-related interruptions also needs to be clarified. The teacher must be in a position to make immediate decisions about whether a situation can be left to resolve itself or whether it requires teacher intervention, whether the problem is mechanical or student-related, whether or not existing rules and procedures cover the situation, whether the situation can be quickly resolved or whether it will require extra time and effort. At issue is the teacher's sense of control in the classroom. Most teachers report that they alter the ways in which they organise and teach lessons to accommodate computer use. Chapter 4 will provide further details.

Currently, teachers are trying to cope with computers in the classroom against a background of uncertainty. They would like to see departmental guidelines to both resolve some of their dilemmas and acknowledge that what they are doing is of value. Most teachers admit that they do not know how computer use develops thinking skills, and that they do not know how to measure such developments should they occur, but they want to continue teaching with computers in the classroom.

Given the present uncertainty about the place of computers in school, the lack of teacher training and the scarcity of resources, the decision to teach computing as a subject may be inevitable for many schools. Teachers who have taught computing as a subject are pointing out the problems and limitations of this kind of activity in which computer awareness is an end in itself. A certain level of computer awareness is necessary if the computer is to be used as an instructional aid, but if we wish to develop the instructional potential of computers in the classroom we need to look beyond minimal competency or awareness as the major goal. The instrumental or tool potential of computing should be addressed early, when computers are introduced, so that the computer can become a resource for students and teacher rather than an additional obligatory demand in an already crowded curriculum.

In defining computer literacy it is useful to distinguish it from computer science. Computer literacy is not the same as knowledge of computer science. A succinct distinction between the two can be made by suggesting that computer literacy is that part of computer science which everyone should know about. As noted above, those taking the comprehensive view commonly define literacies (e.g. language literacy, mathematical literacy, science literacy) in terms of the layperson's perceived needs. In the same way computer literacy should be thought of as the knowledge, understanding and skills the average citizen needs to have with respect to computers. This implies that students should be taught more than simply how to operate or program a machine. They also need to know how computers can be used productively and what the major consequences of computerisation are for a society. This is why it is so important to encourage computer use in all subjects.

According to the educational computing literature, the objectives of computer literacy curricula range from those aiming to raise general awareness of computing through skilful usage to the possession of broader understandings of the personal, educational, social, economic and political contexts and consequences of that technology in society. The type and amount of knowledge one has about computers determines the potential of that knowledge to be personally, socially and politically empowering.

Computer Awareness

This objective aims to provide the student with some general information about computer hardware, software, vocabulary, uses, history, social, economic and political impact. Students may or may not actually see or use a computer, but if they do, the use is usually limited to demonstrations. Teachers and other instructors rely on a variety of sources of information to create awareness about computing, such as books, films and videos. Technical information and noncontroversial commercial applications are emphasised. Much of the

information which has been found useful in the classroom is produced and provided by the computer industry.

Initially, teaching computing was for the most part teaching computer awareness. This may have been appropriate at the stage when the technology was novel and only few machines were available to be shared by large numbers of students. Many writers concerned with the needs for curriculum aims in computing started off with surveys of what had been taught in the late 1970s and early 1980s. The courses tended mainly to convey information about computers and computing, with little opportunity for hands-on experience. This state of affairs certainly explains the reaction of Luehrman and others emphasising the need for a hands-on approach. Even now most of the published educational objectives relate to computer awareness. Computer awareness is not the same as computer literacy, though it might be regarded as an aspect of it.

The reasons for continuing to teach computer awareness are far more compelling than the more formal rationale on which its introduction tends to be based. These reasons often relate to the desire to make the classroom a more interesting and enjoyable place to be in, and for developing new relationships through what is often a shared learning experience between peers, or between teachers and students. The building of these new social relationships in the classroom is often regarded as the reward for the time and effort invested in teaching with computers. Teaching computing thus takes on an affective, self-concept building as well as instrumental value. It makes a statement to students, peers, parents, educational administrators and others about the kinds of instructors their teachers wish to be, the kind of relationship they wish to have with their students, the classroom atmosphere they wish to establish, and their interests. At the classroom level the specification of instructional objectives will certainly be influenced by the decision made with respect to teaching computing as a subject or teaching computing as a tool to be integrated into all areas of learning.

Ability to Use Computers

Another objective of computer literacy emphasises the ability to use computers, i.e. programming and applications dominate the curriculum. The hands-on approach is stressed, because the aim is to train students to *control* the computer. Programming, word processing and spreadsheets are introduced in the elementary grades and more systematically elaborated in the middle school and high school mathematics and business education departments.

The hands-on approach to teaching computing can enable students to develop considerable computing skills, including the ability to use graphics and word processing. Students become confident in their interaction with the computer. But demystifying what is generally a user-friendly personal computer contributes little to understanding the importance and potential of the technology for the individual. In fact, as noted in Chapter 1, a narrow focus on the technical skills of using computers may lead the students to a false sense of empowerment.

> The technical focus shifts attention away from social questions and portrays computers as something to learn rather than as something to think about . . . The computer is portrayed as friendly and accessible . . . and the user is encouraged to think that all computers, even those in large systems, are friendly and accessible. In this manner, computers are further mystified in the very act of demystification. (Noble, 1985, p. 72)

Making the Computer Part of Oneself

A third type of instructional objective is one which aims to have students make the computer part of themselves, in their school and personal work as well as social environments. The attainment of this objective is potentially most empowering, yet it appears to be the least discussed in curriculum documents and least experienced in schools. It introduces computing skills and knowledge within a broader social context, stresses the implications of computer technology and the empowering effects of such knowledge. In addition to some technical knowledge, students need understanding and knowledge of who controls the direction of computing, for what purposes, for whose benefit and whose loss.

One might argue that it is not easy, and in certain cases intellectually improper, to inculcate beliefs and values about a subject which do not arise from the student's direct experience with the content of that subject. If one were writing about mathematics, reading or writing, there would be little disagreement about this point. For example, parents would be properly outraged if their children were asked to spend four out of five days working on the beliefs and values about the subject of mathematics, and to spend only one day on learning to do mathematics. However much we want our students to remember facts about mathematics, and feel good about it, we know that these beliefs and values will be short-lived if the students go out into the world with poor ability to do mathematics. The same applies to all other areas of the curriculum, including computing. In the future, most members of our community will have very real practical needs for understanding computers.

Computer literacy will become as important as literacy in language and mathematics. Like reading, writing and arithmetic, computing gives the student a basic intellectual toolbox with innumerable areas of application. Each one of these tools gives the student a distinctive means of thinking about and representing a task, of writing his/her thoughts down, of studying and criticising the thoughts of others, of rethinking and revising ideas, whether they are embodied in a paragraph of English, a set of mathematical equations, the simulation of a social process or the development of a computer program. Students need practice and instruction in all these basic modes of expressing and communicating ideas. Mere awareness of these modes is not enough.

General Objectives

As yet there are no generally agreed upon curriculum objectives for learning with computers. As noted previously, the US literature contains various lists of items based on *objectives for computer literacy* (e.g. MECC, 1979; Johnson, Anderson, Hansen & Klassen, 1980; Luehrman, 1981), often without any specification of what the specifics of instruction actually are. Most of the objectives to be found in the published literature are representative of the lower levels of cognitive skills. Objectives covering deeper levels of knowledge and understanding are not sufficiently developed, hence ideas of minimum competency cannot based on them.

The US National Commission on Excellence in Education (1983) recommended that high school students should complete, among other things, one half year of computer science. The Commission suggested that, as a result of taking such a course, high school students should be able to:

> Understand the computer as an information, computation and communication device;
> Use the computer in the study of other basics and for personal and work related purposes; and
> Understand the world of computers, electronics and related technologies.

Some aspects of computer literacy might be achieved by requiring students to take an introductory computer course and/or programming courses. For general educational purposes it might be better to begin with the broader curriculum aiming at general literacy. Such a curriculum will put primary emphasis on the direct interaction between the student and the technology, with a goal to help students not just to cope with the hardware and software but to master wholly new analytic, expressive and problem solving skills.

The comprehensive approach to computer literacy requires that many domains be included in a curriculum dealing with computing. Not all of them will be given equal weight by all teachers -- rather, the teacher will pick and choose appropriate goals for particular students.

At the present time and in most contexts, any list of curriculum objectives will be an evolving conceptual structure. Initial lists may be a smorgasbord of objectives for computer literacy which requires ongoing refinement and updating. In any case, the objectives will require ongoing revision to take account of changes in the technology itself. Whatever list is constructed, it must provide guidance not only to curriculum but to those who wish to assess progress. More specific learning objectives and suggestions for assessment are provided in Chapter 5.

COMPUTER PROGRAMMING AND COMPUTER LITERACY

Programming

Programming is the process by which a computer is made to perform a particular task. Programming involves the creation of a formalised sequence of instructions which can be recognised and implemented by the machine. These instructions, i.e. the program, are in themselves a static entity, but when executed, they result in a useful means of information processing. All programs are concerned, either directly or indirectly, with the flow of information. Data, whether stated explicitly or made an intrinsic component of the program, are used as an input which is then processed or *computed*, to generate an output. All of the functions performed by a computer depend, at some stage, upon a program.

The instructions are encoded into a specific programming language. Different languages vary both in structure and in syntax. The choice of language depends on both the application and the computer for which it is intended.

Broadly speaking, the components of a computer program can be categorised into three kinds:

1 Commands which are responsible for the *manipulation of data* within a system. They perform what is referred to as the actual computation and include reading in values from the external environment, assigning values to variables and sending data to the output device.
2 Commands governing the *flow of control* through a program. While their syntax may vary, virtually all programming languages embody four basic structures: sequential commands or statements which are executed consecutively and once only, conditions which are used to select between different sets of commands, depending upon the parameters specified within them and which can alter the sequence of execution, and the repetition of commands. Frequently groups of commands need to be performed many times during the execution of a program. This is achieved by creating a loop within the program which retains control of the machine until a certain condition has been fulfilled.
3 *Procedures* or *sub-routines*. These are sets of commands forming part of a program which may be used more than once. A procedure is called into action by using its name, and *variable* values can be used to show to what the procedure should be applied.

The Value of Learning to Program

As noted previously, an essential question in operationally defining computer literacy is whether one must know how to program in order to be computer literate. For many people the answer may be *no*. However, this does not dispose

of questions relating to the value of programming as far as general education is concerned. A growing body of opinion suggests that there are substantial intellectual benefits to be derived from learning computer programming. Such basic concepts as *iteration* (i.e. a process which repeats the same series of processing steps, e.g. repeated application of a self-contained *routine*) usually programmed as a loop, *recursion* (i.e. the action of a routine calling itself or being called by another routine, sometimes one that has itself been called . . . until a predetermined value is reached), and similar systematic procedures are more difficult to introduce to students outside the programming environment.

Problem solving skills are vital to everyone, and schooling at all levels should improve the ability of students to solve problems. Learning to program a computer in a suitable language can develop problem solving skills. Properly done, developing a program is a process of defining a problem so that it can be broken down into discrete components, none of which is too difficult to handle, even though the entire problem may be quite complex. Computer programming also introduces students to notions of complexity, interconnectedness, uncertainty and the dynamics of a problem space.

Another useful intellectual by-product of learning to program is developing a technique for *debugging*, i.e. for detecting and correcting errors in a problem solution or program. Because of the nature of the process, learning to program involves making many mistakes and learning to diagnose and correct them. Learning to program thus encourages the development of error detection and correction techniques, and even more importantly, develops what some call a *no fault* approach to making errors. In our assessment conscious educational environment, individuals are usually embarrassed by errors and often attempt to avoid thinking about them. This is unfortunate because it is important that we develop the ability to learn from our errors. Computer programming is an activity which makes a strong positive contribution towards the development of this ability.

Unfortunately, many of the general educational by-products of learning to program are still very much hypothetical. Although many people who have learnt to program can testify to the value of computer programming, systematic research to rigorously evaluate these hypotheses is still lacking. There is, however, considerable support for the view that children who are learning to program establish a different kind of task oriented pattern in their personal problem solving endeavours and in the interaction with their peers when they work together in solving programming problems. Research has shown that these patterns of cognitive processing can carry over into other classes, and suggests that dialogue about problem solving and learning processes can transfer to other areas of learning. This is likely to have significant educational value. Indeed, it may be that it is in communication, rather than (as often presumed) in mathematics, that the computer may eventually make its most important contribution. The connection between computing and communication is very strong. More than a decade ago Seymour Papert suggested that the term *computer science* is a misnomer because 'most of it is not about the science of computers, but the science of descriptions and descriptive languages' (Papert, 1980, p. 100).

Choice of Language

Assuming that learning to program has educational value, what languages should we teach? Many would suggest BASIC. However, BASIC is not really useful for complex programs, nor does it have the characteristics that help develop problem solving skills. Indeed, the convoluted logic usually required to work with BASIC requires users to accommodate the requirements of machines and, in doing so, might discourage the development of desired problem solving skills.

BASIC has become popular for two reasons: it does not require the user to learn a large vocabulary of computer terms and it runs on computers with limited amounts of memory. Because the memory capacity of personal computers is increasing substantially without much additional cost, limited memory no longer serves as a limitation. Furthermore, the greater power of other languages rewards us for mastering their somewhat larger vocabularies. If not BASIC, what language then? There are several candidates, among them Pascal, C, LISP, and Logo. Of these four, Logo might be the best option for schools. Logo dialects are now available for a wide number of personal computers.

The strongest assets of Logo might be that it was actually created in an environment of Piagetian developmental psychology and that it functions as an interpreter rather than as a compiler (i.e. it gives the user immediate feedback, as BASIC does, in contrast to Pascal, which is more complicated to use). Logo is sometimes mistakenly described as a language for children, because it has been used successfully with children. However, it is a fully developed language and a powerful tool. It is discussed further in Chapter 4.

Interactive Software

Although the computer is not always thought of as a language tool, it might actually be one of the most useful recent innovations for developing language. Using interactive software enables students to discuss, hypothesise, predict, debate, test ideas and develop thinking skills in a medium which they find highly motivating.

A few years ago one would have suggested that anyone wishing to use the computer as a tool would have to learn to program, but this may not be true any longer. Today a variety of software packages exist, including hypercard, which are powerful, easy to use, require no programming skill and run on small inexpensive laptop and desktop computers. These include adventure games, simulations, word processing, spreadsheets, graphics packages and other information retrieval and exchange devices.

Adventure Games These are computer programs which are, in some ways, like fiction books. However, an adventure game is planned in such a way that it enables the user to interact with the story. There are many variations of games which differ not only in complexity but in degree of difficulty. Adventure games can basically be categorised into three difficulty levels:

Highly structured games: Here the students have a limited number of choices to make at different stages of the game. Usually quite young children can operate the game because the goals are basic with limited options to choose from. These programs are suited to the lower primary level but are also useful for older students who have had no experience with adventure games and/or computers. Because a limited number of choices are required, these games do not lead to much discussion among students, or to critical thinking.

Partly structured games: The students are able to use a wider range or more open-ended options to reach their solutions to various problems in the game. They are suitable for middle and upper primary grades. These games are more open-ended thus offer more opportunity for discussion than the highly structured games. They require the users to be more self-reliant.

Unstructured games: In these games the students themselves are required to work out all aspects of the game. For example, in some games they may be required to work out the rules and objectives, or how to solve the puzzles without added information. Therefore, this type of game offers an excellent opportunity for promoting discussion, especially for the older, more experienced student.

Generally, the game provides the user with a scenario and then offers a choice of actions. The resultant choice then sets the user on a course with many further choices. More sophisticated games do not even offer choices but rely on the user providing all the input.

Simulations A simulation is a computer program which simulates a real life problem or situation. It enables the user to interact in various circumstances throughout the program. The user's actions have an impact on what occurs during the simulation. These programs enable students to explore real world situations which may be impossible to explore in any other way. Obviously, such programs should not replace excursions, visits to museums and to industry, but be used in conjunction with them where possible. For example a simulation based on finding and feeding animals at the zoo could be used as preparation and follow up to a visit to the zoo.

Word Processing Word processing is a term invented by IBM to refer to the process of creating and editing text electronically. Word processors enable the user to type text and then manipulate it. The ability to easily change text formats and styles makes this type of program attractive to all computer users.

Many people equate word processing with typing on an advanced electronic typewriter, but this is actually a poor metaphor. Both word processor and typewriter use a keyboard to enter information, but after that there are only superficial similarities between the two. The word processor allows the addition, deletion and movement of words, phrases and whole blocks of text. Because it is so easy to restructure text it is preferable to think of text processing rather than word processing. A word processor could be regarded as more analogous to a cassette recorder than to a typewriter because they share the facilities of fast forward and fast backward searches, easy editing, and cutting and pasting. Essentially, they are both tools to manipulate ideas, operating in two-dimensional space, whereas the typewriter works in a linear mode only. The word processor is a tool of considerable power and there are a number of valuable learning activities which can be performed even with the least sophisticated word processing package.

Because students in virtually any subject area or discipline need to learn to write, word processing skills can be of great value to nearly every student. Furthermore, word processing software can support the process of teaching English composition by simplifying the mechanical tasks of correcting errors and rearranging text, thus allowing students to concentrate on the more complex aspects of writing. Used in this way, word processing can effectively introduce students to the computer as a tool and can become a first step towards achieving wider computer literacy.

Spreadsheets Spreadsheet software automates the process of producing reports with columns and rows of information. Almost any application which displays numeric information in a table, and needs row and column totals and/or percentages, may be prepared by spreadsheet software packages. Spreadsheet software is valuable particularly when such tables need to be updated regularly, because totals and percentages can be recalculated automatically. Budget tables which must be revised frequently are natural applications, as are decision making procedures using *what if* experiments. Issues relating to the value of this software as a mediator of cognitive development will be discussed later in this chapter.

Graphics Packages We are living in a world which has become increasingly visually oriented. Diagrams and charts have become common forms for displaying information. In the past, the main limitation to using graphics was a lack of the necessary talent to draw them. Now there are a variety of graphics packages which will create pie charts, bar graphs and scatter plots quite easily on paper as well as on screen. They constitute a growing resource that can make it easy to create high quality graphics without programming skill and at minimal cost. This type of package can be useful in a large variety of domains and courses.

Information Retrieval and Exchange The addition of a communication capability, such as a modem, to an inexpensive personal computer can transform it into a powerful device for the exchange and retrieval of information. It can be used to retrieve information from information banks available to subscribers, to search bibliographies, and to exchange messages such as letters, memoranda and other texts via telephone lines. Information exchanged in such a way can be used with word processing systems to provide a very powerful communication capacity. Being able to communicate electronically with students in other schools, even overseas, has become a reality for students in many Australian classrooms.

INTEGRATING COMPUTERS INTO THE EXISTING CURRICULUM

In most schools where computers are used, the computers are not themselves the primary object of study, but essentially a tool. This raises the question of whether there should actually be a curriculum for computing, or whether the curriculum objectives in content or subject areas should be amended to allow for computer-based teaching and learning.

Because computers are regarded as tools rather than as a topic of study in themselves, teachers using computers in existing courses are sometimes reluctant to devote too much time to the development of students' computing skills, and consequently the result could be a cookbook, key-by-key approach to teaching the use of computers. Although this approach might, in the end, lead most students through the required processes step by step, it involves little more than repetition on the part of the students. Used in this way the computer is merely a medium of presentation, like a special workbook or overhead projector, though perhaps a little more dynamic.

Like all other learning, computing should be intellectually stimulating for the students. However, there is still the problem of how to accomplish this, in terms of both class time and the availability of computer literate teachers. One solution would be for Commonwealth or State Departments of Education to commission staff or outside consultants to prepare small modules that could be used in conjunction with existing curriculum subjects, thereby avoiding the necessity to introduce a special course in the technical aspects of computing.

Minimodule of Foundation Skills in Computing

A module aimed to provide a rapid and effective introduction to computer use in, say, mathematics, science, social studies, writing, etc. would accomplish the basic prerequisite goals in educational computing. Students need to be able to work with a computer's operating system to perform file-management tasks, to

use an editor successfully, and to cope with input and output devices. Teachers must not restrict themselves to the use of readily available software that avoids the use of these skills. A foundation set of skills needs to be presented in a manner that allows the ready transfer of what has been learnt among various computers which students might be expected to use in different contexts and in an assortment of subjects.

The ultimate goal is to make students self-sufficient in the way they use computing tools on a daily basis, equipped to solve problems and to learn how to work with new software and computers. The module needs to work for teachers who are not computer specialists. In addition, the module should impose a minimum of overhead in class time. This might involve some restricted time for independent study and practice. Above all, the introductory experience should be pleasant and inspire enthusiasm.

The Development of Reading Skills

Personal computers are an enabling technology. They help the user to collect, organise, store, retrieve and deliver information. One of the most commonly available information handling devices in the classroom is the word processor. As noted above, the word processor is more than a device to produce fancy printing. Word processing is one of the most open-ended and flexible tools with which students can think about the structure and purposes of language. That the word processor is a tool for writing is obvious, but it can also be used as a tool for developing reading skills. In fact, it can be used totally without writing on the part of the user.

One of the views currently held by teachers of reading is that children should develop writing skills as a way of developing their reading, in other words students should be encouraged to write to read. This statement may appear contradictory, but it is based on the sound learning principle of using the child's own knowledge as a starting basis for development, and for *learning by doing*. An adult who keys in a story which the child tells him/her, and then provides a print-out, produces reading material for the child which is personally relevant. Because the poor reader is reading materials in context, and is therefore more likely to be successful when the context has saliency, this personal relevance encourages the development of reading skills.

When employing this approach to the teaching of reading, the role of the adult is critical. If the adult takes an active part in the writing process then the whole social balance will have changed and the child defers to the authority and wishes of the adult, thus losing the all-important personal involvement and/or the motivation for story production. The adult helper is there as a medium through which the child achieves his/her story goal, not as a judge or assessor. Collaborative writing between adult and child may follow later, when the child has sufficient skills to feel confident of sharing the task with the adult.

The use of specific word processing features such as *FIND* and *FIND AND REPLACE* offer opportunities to practise *reading for meaning* and for more general vocabulary development. These commands allow students to reconstruct the text.

The role of the word processor in developing *information handling skills* and as an introduction to information handling packages may not be immediately obvious. At all levels of education, and in many work situations, there is a frequent need to summarise a text. One interesting reversal of this is to provide students with a summary of text, or with a piece containing only key content words, and asking the students to reconstruct the passage in their own words. The texts written by different students can then be compared or related to the original. The latter activities can encourage students to focus on the structure of text, and to consider what is or is not redundant. Active language games which require rethinking and editing as the writer goes backwards and forwards in the text provide useful experiences in the manipulation of language. The word processor allows for repeated revision without penalty. The non-linearity of such exercises benefits students' language facility, their ability to comprehend and create text, and to generally think more flexibly.

The creative teacher can thus use the word processor in many different ways: as an electronic worksheet for basic skills practice, for the development of advanced reading skills, for the decoding of text, to develop an understanding of story structure, and to encourage planning and self monitoring. The word processor can support reading and writing, of course, but it is also an organiser of thought, a notebook, and a trial ground for the exploration of ideas.

The Development of Writing Skills

Writing with the computer is a valuable activity for individuals of all ages. Papert's description of an alienated writer who moved from total rejection of writing to intense involvement (accompanied by rapid improvement of quality) within a few weeks of beginning to write on a computer makes convincing reading (Papert, 1980, p. 30). Sharples (1985) goes further, offering an explanation of children's use of few of the high-level skills of editing and revision when writing with pencil and paper:

> To be a writer is empowering yet every word that a child forms on paper is confirmation of inferiority. However carefully and neatly a child may write, the result is a poor substitute for adult typeface. If we want children to become adult writers, we should equip them with adult writing tools. (Sharples, 1985, p. 10)

This statement begs the question of how any child ever learnt to write, but it does make a useful point. Children tend to be involved in the technical details of text production, which leaves limited cognitive capacity for the higher level creative, organisational and other more demanding activities of writing. If our minds are

trapped by decisions about spellings or grammar, they cannot be used for the consideration of story, plot or even idiom.

Using the word processor allows the writer to concentrate on the process of writing, i.e. the acts of drafting, editing and revising the text. Daiute (1985) suggests that writing on a word processor is more like talking than writing; in other words, it is an interactive exchange between writer and the tool for writing. Text is entered into the machine, which then displays the writer's ideas for consideration. If, on reflection, things do not look right, modifications can be made. The tool responds and a new, clean version of the student's thoughts appears on the screen.

Writing is a multilevel complex process, in which the student writer's attention tends to be directed to the lowest skill components which pose performance difficulties. The notion that novices attend to different aspects of performance in problem solving and other cognitive activities than experts is well recognised:

> Writing is a communicative action that results from multiple cognitive processes that operate simultaneously, producing text through their interaction. For example, there are processes that draw a letter on paper, and those that select and organise ideas. While an expert writer can operate competently on these many levels, a novice tends to become locked into more local levels, a phenomenon called downsliding . . . (Levin, Boruta & Vasconcellos, 1983, p. 220)

One area of concern relates to transfer of training, the exchange of skills between the two media for writing, from pen to machine and back again. Salomon (1988) has shown that children who have been taught writing skills, such as redrafting of text using the word processor, do transfer those skills to pen and paper production. Our experiences in the SUNRISE classrooms at Coombabah support this finding. When asked to draft and write an essay using paper and pencil, the students proceeded in similar ways as they did in computing, and the products were certainly comparable with those of students who do not learn with computers.

Finally, it must be recognised that all technological innovations evolve over time to reflect the needs of users as well as new developments. Over time technological innovation is cumulative. One change does not always replace another. Rather, innovations overlap to form the complex combinations of *old* and *new* technologies so often encountered in modern society. Many existing technological innovations are under review to revise more suitable and improved applications. This state of flux requires constant monitoring and adaptation of curricula relating to the use of technology.

TECHNOLOGY AS A MEDIATOR OF COGNITIVE DEVELOPMENT

The theoretical framework for this endeavour stems from the work of Vygotsky, who first introduced the concept of *mediated activity* into the psychology of thought and language (Vygotsky, 1978). Unlike other approaches to mental functioning, Vygotsky's *activity theory* views cognitive and motivational processes as embedded within larger activity structures whose goals they serve (for expositions of activity theory see Kozulin, 1986; Wertsch, 1985). Activity structures involve *mediators*, i.e. tools and symbol systems which have deep implications for the way in which intellectual tasks are accomplished. Thus, this theory suggests that the introduction of new systems and tools into learning (or work activities) can be expected to change the intellectual aspects of these activities. According to the theory, however, the nature of these new intellectual demands cannot simply be projected from a study of the tools themselves. The demand characteristics of the tools are not all *built into* the tools themselves. Many of them stem from the way the new tools are utilised, i.e. the functional purposes they fulfil and the way tasks involving them are structured and socially distributed. A cognitive analysis of the impact of new technologies, therefore, must be concerned with the varieties of ways with which such technologies are drawn into ongoing activities.

A concrete example relating to literacy may help to illustrate the difference between more standard approaches to cognitive development and the activity approach. A writing system is readily recognised as a form of *intellectual tool* which mediates a multitude of social practices in a culture, including educational activities, work activities, recreational activities, and the like. Here is one technology that scholars have long agreed has cognitive implications. A long tradition within scholarly disciplines, and more recently in anthropology and psychology, has thought to derive these implications from studies of the properties of writing, such as the fact that writing objectifies language, is composed of units that are not marked off in speech, etc. (e.g. Goody, 1987). This school of thought has put forward claims that the intrinsic properties of writing systems, especially alphabetic scripts, promote abstract and logical thinking among those who master them. Literacy programs have often been built around the understanding that literacy both requires and fosters specialised *higher order* ways of thinking.

Empirical work, much of which was conducted within the Vygotskian framework (e.g. Scribner & Cole, 1981), disputes this postulate. Studies of literacy in various cultural and community settings demonstrate that there is no hard and fast relationship between literacy and cognitive implications. Intellectual implications of literacy are variable, and often contingent on the functions which are being served by writing. If literacy consisted only in rote memorisation of a sacred text, its intellectual consequences would appear to be limited to specific rote memorisation skills (e.g. Schieffelin & Gilmore, 1986).

On the other hand, literacy which serves multiple communicative purposes appears to foster skills in organising and expressing complex information in instructional situations, although it has little impact on memory skills.

The proposition that introducing a writing system into a society has a fixed set of cognitive consequences in all places at all times is an argument in the technological determinist vein. In contrast, the activity theory approach assigns a leading role to

> organisational structure of the society adopting writing (is writing the prerogative of a priestly class or available to people in many social groups?),
> specific practices in which writing is introduced as a mediator (is it confined to private uses or does it figure in trade, government, and everyday life?),
> the individuals who are recruited to literacy in the conduct of these practices (do all participants become literate or do some serve as representative scribes?), and
> the conditions under which they use it (is text easily produced?).

Since in the modern world there is considerable communality in the functions of literacy across various societies, we might expect such communality to be reflected in *like* cognitive correlates. It would be misleading, however, to argue backward from discovered similarities in the consequences of literacy to conclude that these are all inherent in the properties of the writing systems. The properties of writing systems have certain potential effects on social and psychological processes, but the realisation of those effects in turn depends on existing, historically created social and psychological factors. The relationship is reciprocal and not one way.

The above framework, illustrated with respect to traditional literacy tools, can guide our approach to new technological systems more generally and to learning with computers in particular. The general message is that the unit of analysis for cognitive studies of computing cannot be restricted to the technology itself, nor to isolated tasks removed from the context of their performance. Such analyses would provide only partial and possibly misleading information for policy makers, curriculum developers and teachers who are concerned with defining educational goals for the future.

Research conducted overseas and in Australia into the *effects of the computer* on students' cognitive development has too often tended to regard the computer as a single factor of change introduced into a classroom, which is presumed otherwise to remain the same. In other words, the computer is perceived as an independent variable the *net* effects of which can be controlled and quantified. In reality, there are no *net* effects. The introduction of computers into a classroom is far more than a *treatment*. The characteristics and potentialities of the computer become inextricably intertwined, not only with the way students might go about learning and problem solving tasks, but with the tasks themselves and the whole

context of learning and teaching. It is not the features inherent in the tool but how students and teachers use it that determines the effects of computers in education.

Software Can Restructure Cognitive Processes

How can computer-based technologies fundamentally restructure the way humans think? Three examples will be discussed in some detail, and others mentioned, in which software has qualitatively changed both the content and flow of the cognitive processes engaged in human problem solving. In particular, what the student does and when he/she does it -- in other words, the component mental operations that a person contributes to the computer-aided problem solving efforts -- can undergo substantial change in comparison with the operation of these processes in traditional problem solving environments.

Electronic Spreadsheets A first example of software programs for personal computers which can restructure, and not merely supplement and thus boost, mental functioning is the electronic spreadsheet. The screen image of an electronic spreadsheet physically resembles a ledger sheet, with cells organised in rows and columns. But the resemblance ends there. In an electronic spreadsheet, one can place a numeral, a calculation, or a formula in the formula area of any cell, which can subsequently be edited, copied or moved. The results of calculations in the formula area appear as the content of the cell. The most dramatic difference between electronic spreadsheets and static paper spreadsheets is that one can change cell entries and see the repercussions of that change recalculated immediately throughout the total spreadsheet. Many lines of thought can be simultaneously activated in the form of dynamic *living* plans, and their outcome compared in terms of crucial variables. This *what if* property has dramatic consequences for the cognitive activities of, for example, budgeting (and financial modelling) and other forms of hypothesis testing.

Before 1979, in ledger sheets representing financial quantities, formulas relating these quantities and change over time were either recalculated by hand after every change, or modelled with mainframe programs under the control of data processing departments. Executives responsible for financial planning were not directly and personally involved in these operations. Personal computer budgeting has become a highly creative means of generating and testing various scenarios in complex financial situations for *what could be*, given different hypothetical assumptions. The effort required to formulate such scenarios in the past and to update them regularly made such explorations not feasible, except in limited fashion by mainframes controlled by data processing departments, not by the executives themselves.

In terms of the restructuring metaphor, the tool (personal computer) has altered the mental work of budgeting. The task has changed: now the predominant component mental operations for the financial planner are planning and

hypothesis testing by means of interactive development and testing of different models for budgets. The temporal sequencing of mental operations in the functional system for budgetary thinking has also changed: now the planner can opportunistically and flexibly test hypotheses in the model virtually wherever and whenever he/she wants. For example, any hypothesis on relationships between cells can be tested by modifying formulas and observing the recalculated results.

Beyond the quantitative increase in efficiency (some estimate saving ratios in budgeting to be 80:1) business planners now run vast numbers of complex experiments of hypothesis comparison and they can include many more variables than they could in the past. They also have a better understanding of the interdependencies of the component operations than before this electronic tool was available.

Furthermore, this tool has qualitatively changed the organisation of budgetary justification and argumentation. Electronic spreadsheets are now commonly used, unlike anything before, to quantitatively justify business decisions in group discussions by on-line comparisons with alternatives. The dynamic *what if* capacities of such a system make it possible to display immediately the consequences of different approaches to a problem that may be suggested during a board meeting of a company.

Finally, at the institutional level, the personal computer electronic spreadsheet has decentralised financial planning. Everyone can *play* with it. The number of mediating links between planning and testing financial models has been reduced rather than increased by the technology, and executives report feeling more in control of their futures.

Problem Solving in Mathematics Similarly to the spreadsheet users, mathematics educators argue that the use of symbolic manipulation programs such as muMATHS, MACSYMA, REDUCE, MILO and MATHEMATICA for doing algebra leads to a profound shift in the functions and structure of mathematical thinking from mechanical operations to problem solving operations (e.g. Arnold, 1991; Conference Board of Mathematical Sciences, 1983; Fey, 1984; Goldenberg, 1988; Grace & Cassidy, 1990; Heid, 1988, 1989; Maurer, 1984a, 1984b; National Science Board, 1983). These personal computer programs and others can easily accomplish the solving of complex numerical and algebraic equations, factoring of polynomial expressions, evaluation of definite and indefinite integrals, differentiation of elementary functions, solution of equation systems, and simplification of equations, even those with radicals (Kunkle & Burch, 1984; Wilf, 1982). What are the implications for how student users of such programs think mathematically? A student using such a program is likely to spend time primarily on algorithm design and search (i.e. solution path finding) of appropriate operators, rather than engaged in the mechanics for calculating numerical expressions.

Consider the task of solving linear equations. Search is not a central concept in algebra instruction today, but a central insight of cognitive science is that

effective problem solving skills in mathematics fundamentally involve search, i.e. knowledge about when to select which subgoals, and in which sequence. In most classroom instruction in solving algebra equations, the teacher selects the operator to be applied to an equation, and the student carries out the arithmetic. The pedagogical flaw in this method is that students do not know when to select the various subgoals (e.g. Simon, 1980) when solving equations independently, even if they know how to execute subgoals (e.g. to do the arithmetic once the operation has been selected).

Algebraland (J.S. Brown, 1984b) is an electronic system for helping students understand algebra by doing algebra problems. The task for the student might be to solve the equation $5(2+x)=20$ for x. A series of *windows* provide the student with a choice of algebraic operators, a solution path record showing all the intermediate expressions, and a search space recording all the steps explored by the student. After selecting an operation and where to apply it, the student can execute it. This creates a second algebraic expression. The search space is represented graphically as a tree which displays solution paths with all the backtracking points and problem solving moves made while trying to solve the equation. The program performs all the tactical, algebraic operations and arithmetic calculations. Students can select the operator and its scope of application, effectively eliminating errors in arithmetic or in the application of operators, and are thus left free for the real mental work of search and operator evaluation.

Operators are also provided for exploring solution paths. There is an *UNDO* operator which returns the equation to its immediately preceding state and a *GOTO* operator (not on the menu) which returns to any previous state. Students can also back up a solution path by applying the inverse of a forward operator (e.g. selecting *DIVIDE* after they have just applied *MULTIPLY*).

Because the windows show every operator used and every state the equation was transformed into, students have valuable opportunities to learn from specific paths of their problem solving, and they can play with possibilities. They can explore the search paths of their solution space, examine branch points on one stem where an operator was used which led down an unsuccessful path, and on another stem try an operation started down a path toward solution.

The above described learning activities are not possible with traditional methods for learning to solve equations. Algebraland and similar software offer new opportunities for different forms and types of learning through problem solving which were not available in static, non-computer-based symbolic technologies for solving equations.

In summary, these types of computer environments emphasise a procedure that is diametrically opposed to the traditional instructional methods. Using this type of software the student chooses when to apply operators, and the computer carries out the mechanical procedures to transform the equation. Students are thus challenged by the problem of search for and discovery of a path of operations that will lead from the initial problem state to the goal of solving for the unknown x.

Learning effective search skills in algebra equation solving is not a trivial task. The cognitive technology of the type of Algebraland reorganises the learning in a way that appears to highlight more fundamental skills to be learnt. It introduces students to the functional system of mathematical thinking for the equation solving task. Similar reorientations are evident in artificial intelligence tutors in the programming language LISP (e.g. Anderson & Reiser, 1985) and geometry proofs (e.g. Boyle & Anderson, 1984). The required component operations are redirected. Calculation of arithmetical operations is eliminated, but students can now analyse and learn from an explicit written history of their problem solving moves in searching for the path of operators. This type of software with its focus on problem solving strategies (as the crucial human component in equation solving, finding geometric proofs, etc.) thus provides students with the opportunity to become familiar with the idea of search, to understand the importance of search in a specific case, and to learn how to improve their search strategies.

The consequences for mathematics education, and for what mathematical thought requires, which result from these new cognitive technologies are remarkable: students need to learn, and can learn among other problem solving skills, how to search effectively. And 'although estimation skills are still central, error-free computation of sequences of operations on numbers and formulas is no longer as important as mental activity in mathematical problem solving' (Pea, 1985, p. 173).

Writing with Outliners and Idea Organisers Two dramatically different kinds of computer-based writing technologies will be described. Both of them are designed to better serve the externalisation and revision of thinking processes facilitated by written language (cf. Pea & Kurland, 1987a).

The first type of tool is an outlining program. It provides a rich technology for interactively creating and revising a structured top-down plan of a written document. Several commercially available examples for personal computers are *Thinktank* (Living Videotext) and *Framework* (Ashton Tate). Their essential property is the capacity they afford the writer of portraying an outline at different levels of detail without revising the content of the document. With this facility one can quickly flip (usually in a keystroke or two) between different perspectives on the document, analyse part-whole relationships, and make and test revisions for their goodness of fit. Teachers using such programs report greater experimentation among students with alternative organisational structures, and vastly more attention during cycles of revision to how the details of the text contribute to the purposes of the whole document.

Notecards (J.S. Brown, 1984a) is a minicomputer tool created at Xerox PARC with a different orientation from the above described outlining program. It encourages bottom-up discovery and definition of relationships among ideas which the writer may have in mind initially only haphazardly, or which do not yield easily to top-down structuring early in the writing process. Through cycles

of shuffling and filing of notecards according to categories the writer can define, one can progressively discover idea structures during writing, which are based on ideas collected from texts and their annotations and linking by various relations (e.g. the rhetorical relations of evidence, comment, argument), which can then be reorganised into a map around which text can be generated. VanLehn (1985) has described in vivid detail his experiences with the powers of *Notecards* as a tool or organising process for the analysis of a complex text. He describes how, by explicitly tagging the nature of relationships between arguments and evidence with Notecards, he found loopholes in the intricacies of his own competitive argumentation for specific assumptions in his highly complex AI model of learning to subtract (VanLehn, 1983).

In both the above described cases, structurally distinctive features of the writing technologies provide the possibilities for reorganising one's writing processes and for trying out different cognitive activities during writing. The closing of the temporal gaps between thought and action, between hypothesis and experiment, which these technologies facilitate, and the rapid cycles of *create-test-revise* which they thereby make possible (much like the bases of spreadsheets and mathematics software) appear to have deep qualitative effects on how problem solving in writing is accomplished. Such processes are not anticipated or captured by the amplifier metaphor of computers and computing.

Other Examples Other examples of computer-based technologies which could lead to the reorganisation and not just amplification of human problem solving processes include:

complex planning aided by project management software and planning programs,
interactive computer programming, particularly in exploratory programming environments, such as InterLisp D (e.g. Sheil & Masinter, 1983),
using computer databases (including icon-based graphic database systems, e.g. Filevision for Macintosh) and graphing software as tools for exploratory data analysis, for organising data, and for framing and testing conjectures of patterns among variables in the data (e.g. Conference Board of the Mathematical Sciences, 1983; Steen & Albers, 1981; Tufte, 1983; White, 1981), and
using simulated *microworlds* to explore principles of Newtonian mechanics (diSessa, 1983) and systems of mathematics (Abelson & diSessa, 1981) in intuitive rather than formal terms.

Further examples, less accessible today to schools because they tend to run on supermicros or minicomputers, but equally dramatic in their cognitive implications for reorganising mental processes are:

- powerful simulation programs, often incorporating highly realistic graphics, for exploring the workings of complex systems, such as electrical systems (e.g. SOPHIE: J.S. Brown, Burton & deKleer, 1982) or physical plants (e.g. STEAMER: Hollan, Hutchins & Weitzman, 1984), and
  AI programs such as expert systems and knowledge-based intelligent tutors.

Expert systems (e.g. Davis & Lenat, 1981; Feigenbaum & McCorduck, 1983; Hayes-Roth, Waterman & Lenat, 1984) are programs which emulate reasoning processes of experts in the field, and are used to support and guide complex problem solving. For example, they can dovetail the decision making processes of humans in medical diagnosis, design of new chemicals, computer-assisted design and manufacturing, automated factories, industrial scheduling, etc.

Knowledge-based intelligent tutors (e.g. Sleeman & Brown, 1982) build detailed models of student understanding and embody in their interactions a theory of tutoring. Issues concerning the broader relevance of these types of cognitive technologies for the future of human learning and development are discussed by Pea (1985).

In the above described examples, computer technology has come to provide cognitive power tools which can improve certain cognitive processes in such significant ways that, once the tool is understood and used regularly, the user feels bereft if it is not available. The computer has opened up new possibilities of thought and action without which one comes to feel at a disadvantage. For an increasing number of people computing has become an indispensable instrument of cognitive activity, and not merely an occasional tool (cf. Minsky, 1983; Simon, 1977).

Software can offer far more than an enhancement in the efficiency of mental operations or an increase in problem solving skills. The quantifiable products of human problem solving have indeed been enhanced, as even the amplification metaphor would lead us to observe, but the software has also restructured the thinking activities involved in such a major way that computer users come to develop new methods of thinking about their mental tasks and discover not previously thought of ways of using the computational tools. Thus, there are emergent properties of computer-aided thought that are unrecognised if one only subscribes to the amplification metaphor of computing.

Deeper Consequences of Cognitive Restructuring through Computing

On the whole, education has not accommodated itself to the strong benefits of these latest technologies. Instead, it has tended to assimilate the computer to its traditional fact-oriented agenda. For the most part, computers are not being used to extend and redefine the student's powers of thinking and expression. A major reason for the prevalence of fact-oriented computer-assisted instruction in schools

today is probably a commitment of the majority of educationists to the amplification metaphor of computing. Where efficiency and speed in achieving already defined and easily measurable educational objectives are the goal, drill and practice software (offering more exercises in less time) is a logical choice. Although a number of educators have begun work aimed to remediate this situation, less effort is being devoted to thinking about the ways in which computers can help serve as cognitive technologies to restructure both the cognitive processes of students (and teachers) and the broader context of the educational environment. Many schools now aim towards computer literacy of their students, but learning and teaching is often *about* computers rather than *with* the computer.

What alternatives are there? Before attempting to find answers to this question, the present context must be examined briefly. The restructuring perspective of educational computing, unlike the amplification perspective, is non-committal with respect to whether the consequences of restructuring of mental activities are positive or negative, developmental or regressive. Here, as in the study of child development, developmental progress is separate from the march of time. Development is an evaluative concept, not a descriptive one. In contrast, the amplification metaphor seems to carry with it the idea that faster and more efficient is better, i.e. the technology offers a means which is regarded as being more adequate to the task in hand.

The restructuring perspective is more problematic. How do we want the effects of computing to manifest themselves? What shape do we want these effects to take? Television, for example, has opened up new global channels of visual communication and tremendous educational potentials. At the same time, some believe that this medium has hampered written language literacy because so much of the children's time is spent listening and viewing rather than engaging in other literacy activities. Similarly, Plato's familiar critique of written language in the *Phaedrus*, which suggests that the technology of writing will weaken people's memories, makes clear the dark side even of a most important technological advancement. Both the positive and negative outcomes of a new technology must be considered.

Thus, it is important to go beyond the recognition that cognitive technologies can restructure mental functioning to arguments supporting specific ways in which they should do so. Such arguments must be theoretically and empirically grounded in our best guesses and our best psychological analyses about what our students will need to know about and do with computers over the next two decades, and ideally during their lifetimes.

Education, whether formally or informally acquired, is by its very nature a moral activity, in which choices are made to direct the paths of learning to socially valued goals. What should be the deeper aims of learning and development in computing, and how can education support these processes?

Which of our current learning objectives (many of them historical remnants of curricula defined in the 19th century) are still valuable and which ones are not?

There are some aspects of our students' world that demand our attention, and that appear to warrant a novel approach.

Looking to the Information Society What should education be like in the information age? We and our students now live in a society which is increasingly dependent on computer-stored information and knowledge, on the use of computational tools for transactions with that information, and the requirement to understand and manage its complexities. A defining feature of this new society has been the information explosion. Knowledge obsolescence is a problem in most fields, and government and private industry need to spend millions of dollars to re-educate employees. Herbert Simon (1987) has pressed the point that in this information age *knowing* has become redefined as a verb describing access to knowledge rather than of possession of information. To know is no longer to have knowledge in one's own memory, but to be able to effectively search for, find, and use the information one needs for particular purposes.

This paradigm shift has profound consequences for the goals of schooling, for the emphases of curricula, and particularly for the creation of appropriate roles for computer-based cognitive technologies in learning and teaching. Although the current uses of computers in education are leading to documentable restructuring of both mental activities and the contexts of learning, they are often unproductive when measured against the criterion of helping students acquire transferable knowledge and skills which will be useful in different contexts and/or over a long period of time. This is why we need to analyse our values with respect to education. Which explicit educational goals are most central, and with respect to which purposes? Answers to such questions are necessary to inform the choice and design of cognitive technologies for education.

With our predominantly fact-oriented curricula, we are hardly preparing our children for the life-long learning the information age requires. Regardless of our media, our aim should no longer be the hopeless task of pouring streams of facts through a straw into the child's memory well, in the hope that the well-bucket will come up full with what is needed. Instead, we can work to help students learn for themselves how, where and when to seek out, organise and *use* information for different purposes. With this orientation, education becomes a process of enabling independent, critical and unique thinkers to take initiatives individually and collaboratively to pose and solve problems, and to apply and develop their learning and thinking skills while accomplishing required tasks. What is required is that we assemble a new vision for education in an age of technology that recognises and takes account of the causal powers of the individual (cf. Harre, 1984; Harre & Madden, 1975). It appears that knowledge of facts will still be useful, but as usable materials about events and problems and to help guide actions, not as ends in themselves nor as inert memory entries to be accessed at the time of assessment and then forgotten.

Emphasis on Cognitive Skills An explicit cognitive skills emphasis is central to the issues considered above. For the reasons described, it seems that a productive approach for cognitive technologies in education will begin to

1 define the cognitive skills children will require, so that they can begin to be in control of their own learning and information management, and
2 design and create new technologies to help support the attainment and use of these skills.

The learning of such skills would thus become more explicit rather than a tacit objective of education, as many ideas in educational computing have been in the past. Among other aims which I see as central in the forms of information literacy (rather than restricted to computer literacy) called for today are:

A strong emphasis on cognitive skills relating to information management, rather than acquisition (e.g. Hawkins, Mawby & Ghitman, 1987), including the formulation of questions and the posing of problems, flexible strategies for information retrieval, information schematisation and inference, information synthesis and integration.
A renewed emphasis on written communication and critical inquiry skills, including the evaluation of arguments as well of sources of information and claims to knowledge.
Metacognitive and self-regulatory skills such as planning ahead, comprehension monitoring and evaluation, cognitive resource management or control (e.g. Schoenfeld, 1985), and learning how to learn (e.g. Dansereau, 1985; Weinstein & Underwood, 1985).
Strategies for creative thinking and inventive problem solving (e.g. brainstorming, problem decomposition, hypothesis formation and testing, and debugging approaches to a task) and systematic decision making methods (e.g. decompositional approaches to comparing utilities of choices, such as cost-benefit analysis) which can crosscut knowledge domains.
Peer teaching and cooperative group problem solving, and the practice of negotiation skills.

Why are these types of skills important? They are important because they appear to characterise the cognitive performances of expert problem solvers in many disciplines, as the AI and cognitive science literature attests (e.g. Barr & Feigenbaum, 1982; Brown, Bransford, Ferrara & Campione, 1983; Greeno & Simon, 1988) and because they are high-yield skills which can be expected to be useful throughout the life span, unlike the traditional fact-oriented curriculum. These broad sets of skills can also crosscut the too often segregated domains of the traditional curriculum, and one would hope that new cognitive technologies developed to support them could be used throughout schooling.

A skills-oriented approach does not mean, however, as some thinking skills programs assume (e.g. de Bono, 1985; Feuerstein, Jensen, Hoffman & Rand, 1985; Whimbey & Lochhead, 1980), that these skills can be effectively taught (i.e. for subsequent use) without strong emphases on domain-specific applications. Method without content is ineffective. Schoenfeld's research (1985) on teaching and student learning of general heuristics such as *draw a diagram* for mathematics problem solving makes this clear. One must ask: what kind of diagram? Similarly, Soloway (1988) demonstrates the centrality of domain specific knowledge for learning general problem solving heuristics for writing Pascal computer programs, such as *break the problem into parts* (which he refers to as Descartes' *divide and conquer* heuristic). He finds that without prior experience in solving problems in a specific domain it is very difficult to identify the subproblems into which one would break the problem. The application of the general heuristic needs to be guided by its prior historical applications in the specific knowledge domain under consideration.

It appears, thus, that general skills can be an instructional goal, but that they must be learnt through content-driven examples (cf A.L. Brown, 1985; Glaser, 1984). It seems quite likely that effective computational tools can be devised for learning and practising such skills through problem solving across different content domains.

Software to Promote Transferable Cognitive Skills Many forward-looking educators and schools have begun to help students acquire the thinking tools used by adults to solve problems in such disciplines as business, history, mathematics, and science, e.g. software for graphing, database management, word processing, and spreadsheet software. The difficulties of integrating adult versions of these tools (i.e. programs designed for different users and different purposes) into the curriculum have become obvious. Versions of these tools which are specifically designed for children have begun to appear during the 1980s, including the widely used *Bank Street Writer* (Kurland, 1987) and the *Quill* writing system (Rubin & Bruce, 1988).

For example, in school studies conducted by Char and colleagues from Bank Street College, New York (Char, Freeman & Hawkins, 1985; Hawkins, Char & Freeman, 1984), it has been found that the powerful information handling tools provided by database management programs require new skills (in problem definition, planning for searches of the databases, etc.) which many middle school students have not yet acquired, and that even some highly creative teachers who deeply value critical inquiry and information literacy are unsure how to teach these skills. How can technologies for education serve not only as tools for thinking, but as tools for thinking skills to develop?

Currently, there are no computer programs available which explicitly aim to tutor the development of thinking and metacognitive skills which are so important for life-long learning and problem solving. Although curricula for thinking and problem solving skills, such as those of Venezuela's Project

Intelligence (Herrnstein, Nickerson, de Sanchez & Swets, 1983), which was developed with the assistance of Harvard University and Bolt, Beranek & Newman, have proliferated in the 1980s (see reviews in Nickerson, Perkins & Smith, 1985; Segal, Chipman & Glaser, 1985), we find no computer-based system for achieving these aims.

Several projects under way at Bank Street College, New York, may contribute to visions of what is possible. In one, Pea and colleagues are building and testing software tools for helping children engage in critical inquiry and construct a personal perspective about various topics, particularly in science, throughout the curriculum. In a second project the same group is building and testing a software environment to encourage the development and use of systematic decision-making skills, including problem definition, analysis of alternatives, evaluating attributes of alternatives, and various heuristics for comparing choices. Paramount in each case is the creation of both effective and enjoyable tools for *learning by doing* and student understanding of how to proceed, which will transcend the specific problem domain under study. The belief of Pea and his group is that if they create useful tools for thinking in these ways, the new visions of education described earlier will at least become possible because they are technically feasible.

We require cognitive technologies for education which embody an explicit knowledge transfer architecture, i.e. transfer activities are part of their very structure. Pea and others are exploring this approach to instructional design in a current research and development project on cognitive skills. In the design of IDEA (Integrated Decision Envisioning Aid), a specific domain of decision making -- family planning -- is used to introduce generalisable aspects of systematic decision making skills (e.g. goal monitoring, constraint planning, defining the space of alternative choices, analysis of attributes of alternatives, plan evaluation and monitoring). Multiple examples of the application of each targeted general decision making method are provided by the software. In this way the learner can at any time explore or be guided to learning generally useful aspects of methods which he/she is learning to apply in the specific case. One might expect that by combining the functions of a domain-specific problem solving tool with those of a general thinking skills coach, an effective program for learning complex thinking skills will emerge.

CONCLUSION

We need to design and engineer environments for the transferable learning that an information age requires. More specifically, to inform education effectively, theory and practice will need to be unified through the intervention of research-informed electronic learning systems which can be used in educational settings. As Greeno (1985) argued: 'Important advances in instructional technology and in basic cognitive science will occur as an integrated activity' (p. 2).

Research and development activities can be united in the creation of educational software prototypes and prototypes of curriculum activities, which are designed and built by interdisciplinary teams of researchers, educators and software developers, and progressively modified in response to formative testing with students. These prototypes can provide sophisticated learning environments for students and simultaneously serve as research tools for determining how skills and knowledge develop with these new cognitive technologies. I would argue that such technologies might serve as the educational infrastructure linking information processing research to educational practice, which Champagne & Chaiklin (1985) suggest is necessary for cognitive science studies to have significant classroom applicability.

Some readers may disagree with the emphasis on the positive effects of computers as reorganisers of mental functioning. The reason for highlighting this implication of computing is that I believe that in the absence of prototypes guided by positive visions of what could be, it is unlikely that we will ever learn what education can become. Just as a child needs tools to think (Papert, 1980) as he or she learns to define and solve problems, so do we, as we work to reshape the aims and methods of learning and teaching with computers, in response to the challenges of an information society. We need to create a plurality of prototypes of electronic learning environments to work with, whose effects, positive and negative, can be empirically examined, reshaped, reassessed and debated, rather than the armchair inspired critiques of computers in education that have tended to overemphasise the long-term benefits of currently available software.

John Dewey (1915) criticised American education at a stage when it had yet to adapt to the changes brought about by the Industrial Revolution: 'The primary waste is not money or resources but human life, the life of the children while they are at school, and afterward because of inadequate and perverted preparation' (p. 59).

As in Dewey's days, we are now in need of fundamental change, guided by research on student learning with emerging cognitive technologies and by communal dialogues about redefining educational aims. Everyone is a stakeholder in this enterprise of reform. Students, teachers, parents, researchers, industry and business, and policy makers all stand to gain or to lose. Working together to shape the technologies which will reorganise human thinking, we may be able to create a new system of education which addresses and fosters the creative spirit and flexibility of the human intellect, that builds on and discovers new worlds of cognition, action and play, made possible by the remarkable symbolic powers of computers, and that yields resilient adults who are ready to meet future worlds more radically different than we can even begin to imagine at this stage.

4

Teaching and Learning

What type of learning environment does one create to maximise the opportunities for learning and personal development of students who have their own laptops or have ready access to desktop computers? It may seem somewhat impertinent to include a chapter on aspects of teaching in a book which is likely to be read by experienced teachers, but discussions with teachers from schools in a number of states have shown that working with computers forces teachers as well as students to reconsider many of their existing ideas about learning and teaching, as well as about the relationship between students and teachers.

When personal computers are introduced into a classroom, they are viewed as a tool for learning. It is expected that learning with computers will allow students to take more responsibility for what and how they learn. Some people might interpret this as an extreme form of discovery learning. They believe that it is the teacher's job to set up the hardware and provide some occasional maintenance of the equipment, give students basic instructions on the use of their laptops, and that from then on learning will take care of itself. It is not difficult to see why this view prevailed. As was noted in previous chapters of this book, the philosophy of Logo is child centred. It emphasises learning much more than teaching. Students are expected to work on their projects and teachers are advised to consider their interventions carefully. But does not mean that teachers should not intervene at all, or that the teacher is unimportant even in the Logo classroom. On the contrary, the teacher's role is vital, and it involves teaching and not just managing the classroom. It is certainly appropriate to give information or to suggest a particular action for some students. Different approaches will suit different needs and situations, so what follows will not suggest hard and fast rules. What is clear is that 'computerised information storage and retrieval is capable of offering liberation from *cluttered brains* and thus giving freedom to concentrate on the development of flexible thinking skills' (Chandler, 1984, p. 56).

The teacher plays the strongest part in the creation of a computing culture for his/her classroom. This culture needs to comfortably support all the students and the teacher. Teachers who are planning to teach students with computers need to examine their own attitudes about computers in society, personal computing and computing in the classroom, acknowledge for themselves the areas creating distress and identify the areas which they believe can be addressed with

optimism. Major prerequisites for the teacher are enthusiasm, knowledge, some experience, hardware and, most importantly, time.

One theme which will run through much of this chapter is that of control. Traditionally teachers have control over almost everything that happens in the classroom, from deciding what is learnt, and in what order, to organising seating arrangements and rationing paper and pencils. Using computers provides an opportunity to encourage children to take responsibility for their work. Handing over, sharing and accepting control can be difficult for teachers and students. In this chapter an attempt is made to highlight some of the special features of the teacher's role in teaching children with computers which can help the process of sharing control. Some of the suggestions made may already be part of the reader's teaching style, others may be less familiar. We shall first consider some philosophical points, briefly introduce Logo and then turn to issues relating to classroom organisation and the interaction with the students.

PHILOSOPHICAL ISSUES

Piaget (e.g. 1952, 1954, 1973) stressed that the principal goal of education is to create men and women who are capable of doing new things, not simply repeating what other generations have done; men and women who are creative, inventive and discoverers. The second goal, he suggests, is to form minds which can be critical, which verify, evaluate and not just accept everything they are offered. In other words, Piaget asks that education should produce independent thinkers and learners.

What Is Independent or Self-regulated Learning?

The terms *independent* learning and *self-regulated* learning are regarded as synonymous here. Both imply that students themselves take charge of their cognitive efforts, and that to a large extent they manage their cognitive skills, abilities and motivation. Independent learners are motivated to succeed and/or to avoid failure. Independent learning combines cognitive strategies, knowledge, skills and motivational states in ways that develop coping tactics and thus preserves feelings of self-worth in students. As they become more independent, learners develop confidence in their learning and problem solving abilities. Confidence in the ability to regulate their own learning enables individuals to attack challenging tasks and to persist in the face of difficulties. This confidence distinguishes mastery-oriented students from students who avoid failure by being passive or defensive about learning tasks; it also helps to establish motivation, to take risks in problem solving, and to expend the effort and perseverance necessary for difficult tasks.

Independent learners approach tasks strategically. From a repertoire of previously acquired knowledge and learning strategies they have learnt to select those which are likely to be appropriate to a particular task and situation. They can evaluate tasks, plan various options and modify their cognitive strategies. In sum, independent learners are aware of effective learning and problem solving procedures (because they have prior experience in their use) and are able to take control of their actions.

Another way of describing the characteristics of independent thinking and learning is as a set of *cognitive preferences*, or characteristic ways in which an individual conceptualises and deals with his/her environment. Independent learning is a way of approaching tasks, issues, etc. -- a preparedness to organise information and experience for oneself, and a conviction that certain strategies are important, effective, efficient, worth some extra effort and instrumental to success.

Conceived in this way, independent thinking and learning become a set of information processing *habits*. Not simply habits in the technical sense of learning theory (as they are not directly responsive to behaviourist principles of acquisition and extinction); instead they are more generalised habits of thought; not just a tendency to use specific behaviours that have become relatively enduring or automatic through repeated performance, but rather the enduring structural and functional bases for such behaviours. Defined in this way, independent or self-regulated learning involves both dispositions and abilities/skills, which teachers encourage and facilitate in their students.

It is now generally accepted among educationists that learning is a continuing process of information acquisition, transformation, association, storage, retrieval and evaluation. Continual interactions and restructuring of perceptions, knowledge, experience, emotions, motivations and interests take place within the individual as he or she adapts to the demands of the environment or adapts the environment to his or her needs.

All human beings have the power to further develop their intellectual strengths throughout life. However, there are two prerequisites for this:

1 The individual must be motivated to continue learning and to exercise his or her intellectual abilities, and
2 The individual needs to have the basic knowledge and practical skills in how to go about these cognitive activities.

Teaching is one of the most powerful mechanisms for facilitating and developing independence and self-regulation in cognitive behaviour. The meaningful communication with adults, peers and the broader environment is as essential for the development of both these prerequisites as for general intellectual growth.

Goals and Assumptions

Teachers foster independent thinking and learning by encouraging students to participate more actively in the communication process of learning/teaching and by guiding them towards assuming control of *how* they (the students) process information.

The goal of such teaching is based on assumptions which differ somewhat from the assumptions which have traditionally provided the framework for the teaching of subject content. Here are examples of the assumptions made by teachers who encourage independent thinking and self-regulated learning in their students:

There are large individual differences in abilities, skills, interests, motivations, thinking and learning strategies in the students in the class; what works for one student may not work for another.

Students may ultimately learn to learn and think independently, but not merely because the teacher taught them. In a very real sense, students must teach themselves, i.e. they must find out for themselves what method of problem finding, problem solving and learning work for them. All the teacher can do is to provide every possible means to facilitate the construction of knowledge by the students for themselves.

Some teachers are so preoccupied with the evaluation of learning outcomes (i.e. *the correct answer* attitude) that they ignore the processes which are taking place during problem solving and learning. In independent learning it is the way the student goes about a task and the related thought processes that count. Very often there are no immediately scorable answers. Ultimately, students who think and plan well will be in a position to generate good answers. However, good answers are not necessarily the result of good and/or independent thinking.

For centuries *thought* was regarded as something that originates in the individual's *mind*, i.e. *inside* the individual, and is then expressed socially. More recently we have come to recognise the importance of social interaction in the formation of personal ideas. Thought emerges, to a large extent, as a social process and is internalised by the individual only after it has been expressed socially. A substantial portion of our ability to think thus originates *outside* ourselves. Interaction and interchange of ideas in discussions with others is essential. This is true at all levels of development. Learning is about communicating ideas and restructuring personal knowledge, attitudes, etc., as a result of such communication.

- To enhance independent thinking and self-regulated learning we need to serve not strictly as teachers, but as models and facilitators, or as *communicative learners* (Black, 1988). We must recognise that we too are learners, and that we encourage independence in our students by fostering and taking part in collective efforts to identify questions and possible answers.

> Class discussion is more than a peripheral part of the curriculum. To the contrary, class discussion is a legitimate and integral end in itself, because it is in such discussion that ideas are produced, shared, reflected upon and internalised. The intellectual development of the individual is enhanced as a result of the group's producing a best possible collective product. How often do we become aware of how difficult it is for members of a group to identify who first had which idea?

Hence, to accomplish the goal of enhancing independent thinking and self-regulated learning in our students we need to abandon some of the assumptions of traditional instructional design, which is rooted largely in behavioural theory, in favour of new designs, which tend to be rooted in cognitive theory. We abandon assumptions such as:

> The teacher is the trainer and the student is the learner.
> Learning is a task for the student, and (in school) only for the student.
> The only thing that counts is the correct answer.
> Class discussion is primarily a means to an end.

In stressing the social and communication aspects of learning, learning to learn and training for cognitive independence, I do not wish to give the impression that the teacher's role is in any way diminished. The aims may have changed, but if anything, more is required of the teacher.

A dominant focus in learning and teaching is on changing students' perceptions of aspects of the world around them. The way in which they think about particular phenomena is at the core of education. However, learning in any discipline involves more than the student making sense of his or her personal perceptions and experiences; it also involves being initiated into *ways of seeing* which have been established and found to be fruitful by the cultural, social, academic or scientific community. Such *ways of seeing* cannot really be *discovered* by the learner -- and if he/she happens to come across or hit upon such consensus viewpoints he/she would be quite unaware of the status of the ideas. It is the teacher who introduces ideas and viewpoints from the outside world of learning into the class discussion. Even more importantly, in most situations only the teacher can provide feedback as to the *consensus* and *status* of these ideas.

Meaningful Tasks

A central theme in Dewey's writings (e.g. 1902, 1938) on education is the notion that classroom activities must be related to the child's experiences, interests and goals. This was a radical view for an era in which didactic teaching was the most acceptable method of instruction. Although the general notion expressed by Dewey has found wide acceptance in Australia and other Western countries in

recent decades, many teachers are experiencing difficulties in implementing it, because of limited resources, materials and training. It is an expectation for many people in the field of educational computing that the personal computer can be a resource for engaging children's interest and fostering more active, creative and independent learning. This expectation is based on two assumptions, namely that children are intrinsically motivated to work on tasks which are meaningful to them, and that the most effective educational environment is one which provides meaningful tasks, that is, tasks which embody a function or purpose that is understood by the students.

While some children enjoy learning about a particular topic for its own sake, in most cases facts and skills are best learnt in connection with wider concepts and ideas which give them meaning and significance. In this way, not only are students motivated to master the facts and skills, but they have a framework in which to understand the logical, scientific, technological, social or cultural significance of the facts and their relationships to other facts.

The above assumptions leave two fundamental questions unanswered:

1 Where do the goals which interest the students come from? Are they inventions of the students, or are they imposed by the teacher?
2 What is the relationship between the goals which students work towards in the classroom and the tasks with which they will be confronted in the world outside school?

Dewey regarded the extremely child-oriented approach as as unsuitable as the traditional view that the teacher must impose the classroom tasks. The teacher has very important responsibilities, which include suggesting tasks and presenting to the students alternative interpretations of problems. In many respects Dewey's approach is more consistent with the socio-historical approach to child development described in the writings of Vygotsky (1978) and Leont'ev (1981), in which the importance of the teacher/student interaction is emphasised, than with the universalist approach of Piaget which de-emphasises the cultural context.

Meaningful tasks may come from a variety of sources. One source is the pool of spontaneous ideas children themselves have. Most children have one or more topics which they simply *like*. However, for most topics in classroom learning, this source may not be the most important. Teachers can make classroom tasks meaningful by showing students their significance in terms of a variety of uses for the skills involved. The functional learning environment created in this way can be a simulation of the real problem (e.g. role-playing a business transaction as a context for doing arithmetical calculations), or it can be a real problem (e.g. actually having a stall at the school fete to raise money for another computer). The functional learning environment can also be of a more abstract nature (e.g. a geometry problem can provide a meaningful context for calculating the size of an angle). A teacher can create interesting functional learning environments by

crossing traditional discipline borders (e.g. by showing how geometric concepts such as triangles can be used in geography to solve navigation problems).

An approach to the second issue, i.e. the relationship between classroom and real world goals, is closely related to the first. It would be reasonable to suspect that transfers of learning from one domain to another and the usability of school learning in later life are inseparable from the variety of functional learning environments in which they are embedded. Being able to see the same fact from multiple perspectives (e.g. recognising the different uses that can be made of a tool) engenders a flexible approach to acquiring knowledge that would otherwise be absent. This flexibility makes it possible to adapt knowledge to new functional environments that cannot be specifically anticipated in the classroom.

Personal computers can play a useful role in functional learning environments because of their capacity for simulation and because they themselves are important tools for the solution of a variety of interesting real world problems. Computers do not function on their own. A teacher must build the bridges between the tool, the school task, the thinking skills, and their functional significance for the culture beyond the classroom.

The teacher's role in educational computing is to provide an environment which is in sympathy with the child's level of development in order that appropriate intellectual leaps can be made as efficiently as possible. The teacher has a far more active role in classrooms operating according to the principles advocated by Piaget (1973), Vygotsky (1978), Leont'ev (1981), and Bruner (1983) than in the traditional classroom. For example, the teacher's role might be considered as one designed to provide a scaffold for children's problem solving (Wood, Bruner & Ross, 1976). Here the teacher is a key interventionist, providing help when the student is in difficulty, standing aside when he/she succeeds, and generally supporting such abilities as to select, remember and plan which might as yet be underdeveloped in the students.

These ideas have had profound influences on the organisation of modern classrooms, and they also provide the basis for an equally dramatic reorganisation of thinking about the use of computers in classrooms. In place of drill and practice, Papert (1980) offers a Piagetian vision of cognitive development driven by interaction not so much with the total world itself, but with a simulated microworld accessed through the Logo programming language.

Constructing Knowledge

It has long been recognised that learning (with understanding) involves the structured organisation of a knowledge system in which concepts take their meaning from the theories in which they are embedded. Central to this perspective is the historically important view that learning comes about through the learner's active involvement in knowledge construction. Within this broadly *constructivist* perspective, learners are thought of as building mental

representations of the world around them which they use to interpret new situations and to select actions within them. These mental representations or conceptual schemes are in turn revised in the light of their *fit* with experience. Learning is thus seen as an adaptive process, in which the learner's conceptual schemes are progressively reconstructed so that they are in keeping with a continually growing range of experiences and ideas. It is an active process of *sense making* over which the learner has some control.

In so far as it views learners as architects of their own learning through a process of equilibration between knowledge schemes and new experiences, this perspective reflects and builds on Piagetian views. It differs from Piaget, however, in two significant ways. Instead of focusing on the development of general logical capabilities, this theoretical position emphasises the development of domain specific knowledge structures. In addition, whereas the emphasis in Piagetian theory has been on the personal construction of knowledge through an individual's interaction with the physical environment, the current constructivist perspective also acknowledges to a greater extent the social processes in knowledge construction both at the level of the individual and within the community of *experts*. The writings of Vygotsky have been increasingly influential in shaping thinking about these social and cultural influences. What is internalised by the child during learning is not what the experts say, but a version of the intera ctions that constitute the joint activity. Thus, without coercion, these interactions guide children towards the cultural interpretation and significance of the tasks in which they are engaged (Newman, Riel & Martin, 1983).

The essence of this school of thought is that the human intellect develops naturally through interaction with the environment. Through this interaction the child discovers the properties of the world and the characteristics of his/her own relationship with the world. The contention is that interaction is of prime importance, because it is the only way in which we can come to understand our personal world and learn how to operate within it and upon it. The reality which we come to understand is, under this theory, a personal construction, and the process of construction is fostered by interactive experience. Cognitive development is seen not as the product of an accumulation of facts, but as being driven by the individual's interactions with the physical and social environment. According to Piaget's theory, learning is under the control of the learner and knowledge and skills cannot be taught directly. Self-directed thinking, learning and problem solving, through actions in the world which teach us by means of feedback which they generate, is the essence of this view of development. It is a view which transfers readily when we consider the impact of computers in the classroom, for here we can provide children with a rich microworld which they themselves can explore with ease and little risk.

If education is to be effective, it must take into account the student's contribution to the learning process. Educators must consider how learners interpret the accepted body of knowledge, including both content and technique. A fundamental principle of cognition is that learning requires knowledge. Yet,

cognitive research also shows that knowledge cannot be directly given to students. Before knowledge becomes truly generative (i.e. knowledge that can be used to interpret new situations, to solve problems, to think and reason, to learn) students must elaborate and question what they are told, examine the new information in relation to other information, and build new knowledge structures. Educators are thus faced with a central problem: how to help students get started in developing their base of generative knowledge so that they can learn easily and independently. Teachers and curriculum developers will have to learn more about what students understand, and then apply what they find out, to improve teaching.

Problems to be solved have to become the students' own. How individuals perceive tasks is influenced by their own generalisations and extensions of the information they are given by others. By the time this point is reached the students are no longer working on the *teacher's* problem; rather they are exploring their own. In short, they are doing mathematics, writing, logic, literary interpretation, etc. The task as given by the teacher may be seen as a springboard for discussion (including amplification) of ideas such as establishing subgoals, working backwards, assuming you have a solution and determining its properties, exploiting extreme cases, solving the problem in more than one way, generalising, and creating one's own problem. It is up to teachers to:

help students understand that a problem is not a problem until one wants to solve it;
build a supportive classroom atmosphere in which students will be prepared to tackle the unfamiliar and not feel threatened when they experience difficulties;
allow students to pursue their own paths towards solution and assist them, when necessary, without giving the answers away;
provide a framework within which students can reflect on (i.e. think about, discuss and write about) the processes involved and thereby learn from experience;
talk to the students about the processes involved in doing and using mathematics, science, geography, writing etc., so that they can build up a vocabulary for thinking and learning about it. Students learn much more effectively when the teacher draws their attention explicitly to the strategies and processes involved.

To attain this goal teachers need not only a clear conception of what is to be learnt but also an ability to see this knowledge through their students' eyes. More specifically, this ability includes the following:

Knowledge of students' typical interpretations of questions, instructions, procedures, and vocabulary at given ages and levels of achievement.
Knowledge of individual children's unique interpretations of these same topics.

- Knowledge about how to introduce formal domain knowledge by building on students' existing abilities, by helping them to generalise informal knowledge to new and abstract situations, and by encouraging the formation of connections between what the student knows and the abstract representations of, say, mathematics.

Constructivism has multiple roots in the psychology and the philosophy of this century. These include the developmental perspective of Jean Piaget, the emergence of cognitive psychology under the guidance of such theorists as Jerome Bruner and Ulrick Neisser, the constructivist perspective of philosophers such as Nelson Goodman. Central to the vision of constructivism is the notion of the organism as an *active agent*, who does not merely react or respond to stimuli as in the behaviourist view, but engages and grapples with his/her context, and seeks to make sense of things.

In particular, learners do not just take in and store up given information. They make tentative interpretations of experience and go on to elaborate and test these interpretations. Even when the learning process appears to be relatively straightforward, for example when learning a short poem or a new word in a foreign language, constructive processes operate. Approximate mental structures are formed, elaborated and tested, until a satisfactory structure emerges.

The main thrust of Piaget's work has been to map out distinct stages of development and to describe ways in which the child constructs a view of the world which either accommodates existing cognitive structures so that they fit with new knowledge, or assimilates incoming information so that it fits existing structures. The view is strongly individualistic and constructivist; the impact of this theorising upon educational practices is best exemplified by individuals engaged in discovery learning. Piaget makes some reference to the significance of collaborative work and the importance of cultural contexts, but these themes were not part of his research. In contrast, Vygotsky (1978) has argued that learning and cognitive development result from a process which is essentially social rather than individually based. The nature of education is to share meanings and interpretations of what happens in the world by an elaborate communication process. Essentially the skilful teacher provides tasks which lie within the learner's *zone of proximal development* and provides enough support to allow the learner to succeed. As a result of assistance received on tasks which lie within the zone of proximal development, the child learns to internalise the processes offered by the teacher, so that the nature of what is learnt, and the cognitive development which results, will be determined by the environment in which learning takes place. Vygotsky also talks about the zone of proximal development in interactions between a child and more able peers.

Vygotsky's approach has been contrasted both with approaches in cognitive science (e.g. Edwards, 1990) and with Piagetian theory (e.g. Smith, 1989). Piaget is seen to offer a biological view of development, and Vygotsky a social view. One might view a Vygotskian perspective as one where the learner is led towards

some view of reality held by the tutor, while the neo-Piagetian view is of learners who work together to negotiate a joint view.

All these approaches share a view that knowledge is socially constructed, and all are consistent with general constructivist views (e.g. Neisser, 1976; Berger & Luckman, 1967) that humans interpret the world around them and build theories (though often implicitly) about all aspects of their lives. At any time these implicit theories shape the way the world is viewed and the way events are interpreted. Constructs about events or people can be changed by evidence, discussion, reflection or direct teaching, and it is almost certain that no two people will see the world identically. Failure to take account of current constructions, and the way they might be modified, is likely to lead to a failure to modify these constructions at all.

If learning has this constructive character inherently, it follows that teaching practices need to be supportive of constructions that occur. The critique by constructivists of conventional teaching practice is that it is not supportive enough of the constructive processes which need to take place in the minds of the learners.

A considerable amount of computer-based instructional material is currently available to teachers, much of it of either the tutorial Computer Aided Instruction (CAI) or Intelligent Computer Aided Instruction (ICAI) kinds. These materials implicitly build on the view that knowledge exists in some absolute sense, that the structures imposed upon the world by current educational frameworks are *correct*, that knowledge is *acquired* by individuals, who accept the structures within which it is presented, that the endpoint is acceptance of some abstract intellectual structure which will be similar across learners, no matter what the original context of learning, and which can be applied to a range of domains. This view conflicts with constructivist ideas and with evidence from cognitive psychology. Ridgway (1988) discusses the associated pedagogic problems.

LOGO

The computing language used in the SUNRISE classrooms at Coombabah is Logo. A strong view exists in the educational community in Australia and overseas which suggests that Logo creates a good environment for programming, and encourages good programming techniques. However, BASIC shares these as well as other characteristics of LOGO, including its versatility and availability. An initial aim of the SUNRISE project appears to have been to answer some questions about the cognitive and social impact of Logo in a Year 6 and Year 7 classroom. An interwoven theme was how student and teacher assumptions and understanding concerning the nature of programming and its requirements changed as they became increasingly familiar with the programming culture emerging in the classroom. In this section we reflect on how our observations enabled us to look more closely at the distinction between the cognitive skills that

might be practised through some uses of formally elegant symbol systems such as Logo and the way in which it evoked particular practices in the classroom.

During the first year of the SUNRISE project at Coombabah (cf. Ryan, 1991), teachers intended the computer activities to be largely child-initiated, so as to encourage the child-centred, Piagetian *learning without curriculum* advocated for Logo (Papert, 1980). While the teachers in the first year gave students some simple instruction in Logo during the first weeks and occasionally held group sessions to introduce new aspects of Logo during the year, their self-defined role was principally that of constructively responding to students' questions and problems as they arose. Students' primary activities were the creation and development of their own computer programming projects.

The second year appeared to differ from the first in that at least two of the teachers decided to take a more directive role in guiding their students' explorations of Logo. These teachers gave more regular group instruction to introduce key computational techniques, and to demonstrate how they work in procedures. Students were required to complete specific assignments which required familiarity with Logo concepts and basic programming skills.

Many educators have been focusing on the use of computers for drill and programmed instruction -- to provide individualised practice and instruction in usual curriculum areas. In the SUNRISE classrooms at Coombabah the teachers have agreed, informally among themselves, on additional aims which involve making use of computers:

- to provide an environment in which learning can be intrinsically motivating and fun;
- to allow students to discover, explore and create knowledge;
- to help develop skills of thinking and problem solving;
- to make some of the most powerful ideas of the developing computer culture accessible and tangible to students at an early stage of their schooling.

A most striking impression the visitor to the SUNRISE classrooms gains is that of the powerful motivation which the computer displays, especially the graphics can create. Every bit of the student's attention is focused on the screen. And this powerful motivation is waiting to be harnessed towards intellectual growth and learning.

What Exactly Is Logo?

Logo is a computer language which was developed to provide an environment which allows learning to take place as naturally as possible. Seymour Papert and his colleagues Bolt Beranek, and Newman, and later at MIT, set out to create a computer language which would combine the capabilities of artificial intelligence

with the theories of Jean Piaget in order to allow a learner to build his own intellectual structures through estimation, interaction, experience and revision.

> Logo is one of the most powerful of computer languages available for personal computers today. The *power* of a computer language does not come from what you can *do* with it. Any program you can write in Logo you can also write in BASIC, Pascal or FORTRAN. Rather the power of a computer language is related to what you *think* with it. Less powerful languages, like BASIC and FORTRAN, force you to attend to lots of details, such as where you must put a semicolon or how long a word can be. Logo has a few simple rules of syntax which are applied uniformly, which makes it easier to focus on the task at hand. (Friendly, 1988, p. viii)

Logo is a list processing language which can be used to achieve a number of purposes, for example text processing, interactive simulation and music production. The language is probably best known for its graphic capabilities. Logo graphics have been used for many mathematical purposes including the acquisition of geometry and mental arithmetic skills, as well as the appreciation of general heuristics for problem solving such as breaking problems down into subproblems. Different versions of Logo graphics are available, and have been extended beyond the production of graphical displays on the screen to more concrete manifestations as turtle graphics. For example, a robot, which can look remarkably like a turtle, drags a pen around a piece of paper on the floor. Instructions to move the pen are given through the computer in the usual way, but the product is directly accessible in the form of a pen and paper drawing produced by the turtle. Lego-Logo is a further extension of this directly accessible manifestation of programming. Robots, small machines, etc. are built with Lego. These are connected to the computer, and their movement or operation is directed through programming instructions from the keyboard.

According to Papert (1980) Logo is an environment in which children can learn fundamental mathematical concepts and powerful problem solving methods without the intervention of teachers. Papert takes his inspiration from Piaget, who has argued forcefully that 'each time one prematurely teaches a child something he could have discovered for himself, that child is kept from inventing it and consequently from understanding it completely' (Piaget, 1970, p. 175).

The Logo language is designed to provide an environment in which self-directed and independent learning are encouraged. The learners themselves should be *in charge* of

> seeing a problem to solve,
> making choices,
> playing with the problem, experimenting and trying out solutions,
> building on what he has already done to do something more. (Harper, 1989, p. 1)

One of the most popular aspects of Logo is that it allows for the creation of graphics effects with repeated sections, such as the petals on a flower, or trees in a forest. To produce a shape, the turtle follows commands to move forward by a

stated distance, and to turn right or left a stated number of degrees. Combinations of commands can be given names, and then be used as procedures within other parts of the program by reference to these names. So, if the student wants to draw a flower with a number of petals, the instruction for a single petal would be given a single name, together with an instruction for moving to the starting point for the next petal, and then in order to draw all the petals with one instruction the student would give a repetition instruction. Once the flower is complete, then all of the instructions necessary for the flower could be given one name, and this single instruction would produce a complete flower. Similarly, a patch of flowers could also be drawn by a single instruction which called upon the procedure for each flower, and so on. The claim is made by Logo theorists that the experience of debugging is of particular benefit for the development of more general problem solving skills.

Evaluation of Logo Effects

Papert and his colleagues claimed that experience with Logo benefits children's cognitive development. As was noted in Chapter 2, attempts to evaluate this claim have brought mixed results. Strong support for the assertion was provided by Robert Lawler (1985) whose book *Computer Experience and Cognitive Development* describes the extensive case study conducted on his six-year-old daughter over a period of six months. Lawler himself acted both as personal tutor and evaluator. He concluded that the effect of this experience with Logo allowed his daughter to demonstrate behaviour typical of a child in Piaget's stage of formal operations, i.e. far beyond the expected attainment of an average six-year-old. The examples of her problem solving, planning and debugging activities which he presents are certainly impressive.

Early evaluations related principally to the Brookline project (Papert, Watt, diSessa & Weir, 1979) and the Bank Street studies (Pea & Kurland, 1983; Pea & Sheingold, 1987). The Brookline project report contains positive evaluations which are themselves difficult to assess, but the Bank Street research found no differences between a Logo group and a control group on a non-programming planning task. The failure to find improvements in planning is important, because this is one of the few direct tests of the claims regularly made for the benefit of learning to program. Finlayson (1984) and Clements & Gullo (1984), discussed in Chapter 2, described clear benefits of Logo experience for the development of mathematical thinking skills. More recent studies have confirmed the positive effects of Logo programming for the early development of mathematical concepts (e.g. Hughes & Macleod, 1986; Robinson & Uhlig, 1988).

Not all evaluations of Logo have found positive effects. Pea & Kurland's review (1984) comes to the conclusion that the idea that programming experience can transform children's minds is itself a form of *naive techno-romanticism*, and after reviewing a number of Logo evaluations Simon (1987) agrees. Whereas

most recent reports (e.g. Underwood & Underwood, 1990) do not support such a scathing dismissal, some caution in the acceptance of all claims by the proponents of Logo is warranted. Certain benefits can be observed in children. These include students' ability to generate creative ideas, the development of spatial skills and improved numeracy. There are studies which have shown positive transfer from Logo debugging to other debugging tasks (e.g. Lawler, 1985; Klahr & Carver, 1988), and also studies which show improvements in specific mathematical and spatial abilities after learning to program (e.g. Clements & Gullo, 1984; Finlayson, 1984; Hughes & Macleod, 1986; Robinson & Uhlig, 1988).

As with many applications of computer-based learning, one of the greatest attractions of Logo is the motivation that it generates in the children using it (Lepper, 1985; Hughes & Macleod, 1986). Mostly this is measured by time-on-task. Obviously, claims for educational benefits must be based on measures that are more profound than the latter if we are to improve the quality of students' cognitive skills and not only their powers of concentration.

A review of the literature relating to uses of personal computers as aids to the development of children's thinking revealed that learning to program with Logo, using databases (e.g. Underwood, 1986; Underwood, 1989; Underwood & Underwood, 1990), and using problem solving games and simulations, can each be seen to produce changes in the ways in which users think about their worlds. There is no curriculum as such in the use of these programs: these applications are educational tools with open-ended uses. Irrespective of the specific educational goal, what is acquired by the student is procedural knowledge. In the case of database and simulation activities some investigators have found sudden and strong developments in hypothesis testing, categorisation and questioning skills of students (Underwood & Underwood, 1990). Most of the gains noted here and in Chapter 2 were observed after only a short period of computer use. Education is a long-term activity, but educational research projects relating to the use of computers in schools tend to look for changes after a few months of experience with the computer. Unfortunately, our empirical study in the SUNRISE classrooms at Coombabah was no exception to this tendency. As Snow & Yallow (1982) have shown, the impact of any one educational treatment may not manifest itself for several years, and equally may continue to show an effect when students have moved from one school to another. It is impossible to decide whether any of the studies reported in the literature would have come up with any long-term changes in the cognitive development of the children who participated. The measures just were not taken. It is unfortunate that so few research projects look for changes over the course of several years rather than weeks or months.

CLASSROOM ISSUES

Teachers faced with the task of integrating computers into learning and teaching for the first time will find themselves in diverse and perhaps initially uncomfortable roles. In addition to more traditional activities, such as being an instructor, demonstrator, evaluator, etc., they will take on such jobs as technician, timekeeper, solver of management tasks, observer, collaborator and model learner.

> The role of the teacher . . . is facilitator and co-learner, rather than the source of all knowledge. There are a number of ways to organise a classroom [with computers] to support this role and produce a suitable learning environment. These ways include demonstrations, peer tutoring and group work. (Queensland Department of Education, 1988, p. 3)

Teachers share in the process of learning with their students. They seek information from their students, observe and document observations, and extend ideas. Two of the teachers at Coombabah have recently reread Papert's work, and they read some journals relating to educational computing, particularly on the use of Logo. On the whole, teachers introducing computing into their classrooms for the first time are faced with enormous time pressures and are finding that the best they can do is just to implement what they know.

Hardware

An obvious aspect of the teacher's role in a computer-rich classroom is that of managing the hardware, i.e. the computers, printers, disk drives, floor turtle, etc. It is quite natural for teachers who are unfamiliar with the machinery to be nervous, and it is important that they allow themselves enough time to become confident. It also helps to show the students how to use all the facilities they need. Technical assistance should be available to teachers at relatively short notice, otherwise much teaching time will be wasted by the teacher in the role of novice technician.

Students will need time to get over the novelty of using the computer, printers, etc., particularly when they are also getting used to the freedom to choose what they will work on. Time will have to be apportioned for this purpose. The printer is a valuable device and it is good to encourage students to print out their procedures, so that they can study them more intensively than they could if they were displayed on the screen. Some teachers report that initially, whenever a student wanted to print out a procedure, the printer was already occupied by other groups using it to get (several) copies of pictures they had drawn. It is perfectly reasonable for students to take away a copy of a screen picture they have designed, but this is very time-consuming. It might be tempting to limit this particular use of the machine, but planning might make it possible to allow things

to progress more naturally. Gradually students will become tired of printing out everything. They will learn to discriminate.

Resources

Among the resources in the classroom should probably be a handbook on Logo, BASIC or whatever computing language is being used, which is written in a way that is accessible to the students. This is important if we want the students to answer some of their own questions independently. Some manuals start with a tutorial section which the reader has to work through. This is not likely to be helpful in the classroom. The aim should be to provide a resource which could be picked up for just a few minutes and then replaced when the required information has been obtained. A home-made booklet with explanations which are short and to the point might be the best idea. An example of the use of a *primitive* need only be given, if its sense cannot be conveyed in any other way. The manual is meant to be used as a reference, either to remind the student of the syntax of a particular word or when a new word looks as though it might be appropriate for a particular task in hand.

In the SUNRISE classrooms at Coombabah the students are given page by page explanations of commands with which they make up their own manuals. As a consequence of this, the presentation of the manual might be quite dry and the students are not as involved with it as they would be if it were more contextual. Students need to be taught to use available resources. For this reason, it might be best not to use manuals or handbooks for the first few weeks of the school year. In the early stages students really need only a few commands, and the teachers can provide the format for the use of these by designing a large poster. Booklets and handbooks can be kept in a cupboard and introduced slowly, i.e. whenever it seems appropriate for particular students. Wall posters can be used to illustrate essential early commands and to display students' work regularly. The latter might include screen dumps together with computer code, to encourage students in yet another way to develop projects.

Curriculum

In planning classroom experiences aimed to develop independent thinking and self-regulated learning with or without computers, it is important to consider the developmental levels of the students, the mode in which information will be presented, and the subject matter that is to be acquired eventually.

As in all instructional planning, learning tasks in computing generally move from those requiring simpler operations to those which are more complex, i.e. from more concrete and observable to more abstract dimensions, and from an emphasis on working with known materials towards creating or inventing new,

previously unfamiliar approaches. Some aspects of independence appear to develop slowly and experientially, but their development is facilitated by tuition and practice. The same applies in learning with computers.

Learning activities can be developed which cover a wide variety of areas. For example Friendly (1988) demonstrates a range of educational domains which can be explored concretely in a Logo learning environment. These include generative grammars, physical laws of motion and mechanics, artificial intelligence and robotics, and the ideas of calculus. Some of these seem to be quite difficult. The important characteristic of Logo as a tool for learning is that it allows difficult concepts to be defined as procedures which make the computer actually *do* the thing which the concept means. The important characteristic of Logo for educational purposes is that it provides the means to create concrete, often graphic, models of learning domains, which can be manipulated by the learner. Such simulations, often referred to as *microworlds*, show how the rules or laws of a particular system work. Students are encouraged to ask *What would happen if I changed the rules?* and can thus come to understand ideas and theories which can go far beyond what would traditionally be expected at school level.

Students can learn new ideas from one another either by looking at wall displays or one another's screens or by listening to peers describing new projects. Sometimes it is necessary, however, for the teacher to introduce a new idea, because none of the students stumbled across it or asked a question which allowed the teacher to introduce it to them within the context of their work. For example, a teacher who was teaching a group of able 12-year-olds with Logo for three weeks found that none of the students were using variables. The teacher did not want to have a formal class lesson, where the students had to listen before they would do some boring exercises, but he did think it worthwhile to bring the idea of variables to the students' notice. The teacher's resolution of the problem was to spend ten minutes without computers, explaining to the class how to write a procedure with inputs. He did not dwell on technicalities at this stage, but stressed how an input made a procedure more flexible than it was before. In other words, the teacher gave the students the ideas which they could use later.

Planning is another activity which is best encouraged away from the machines. Professional programmers probably spend more time with paper and pencil than they do at the keyboard. Students need to be encouraged to plan their work, but there are dangers in overemphasising this. The students are learning to program and so they will need to make mistakes. It is often easier to explore different possibilities at the keyboard. When students learn new techniques they need to practise them before they can use them efficiently, and this is not possible without the computer. We have seen students spending a long time planning a drawing in their exercise books only to reject the plans altogether when things went wrong. However, it is useful for the students to do a quick sketch before they start to type. The teacher should make it clear that the children do not have to be bound by every line of their sketch but that it makes working much easier. It is not uncommon to see two children having a debate because they are working

from different mental images of what it is they want to draw. Their sketch will focus their collaborative activities.

Lesson Format

Organising lessons in which students are working on computers will involve making considerable changes to what might be the normal classroom routine. This is equally true for lessons in a computer room, in a classroom with a limited number of machines or in a situation in which every student has his/her own laptop.

Clearly, learning with personal computers is not compatible with traditional didactic methods of class teaching, but fits naturally into a situation where students are working on their own and/or in groups. Handling a lesson where students are working in this way does present quite different problems to those of talking to the whole class. Many teachers have found that there is a danger of getting caught up with the problems of one group at the beginning of the lesson, so that other groups do not settle down properly. It can be helpful to make a deliberate effort to deal only with immediate problems in, say, the first ten minutes of a lesson, until all student groups are settled and able to get on with their work. Often, the initial problems in any particular lesson will be technical ones which might or might not be dealt with quickly without absorbing too much of the teacher's attention. Getting students settled and working quickly is important in order to establish a good working atmosphere, which will later allow the teacher to spend longer periods giving more concentrated attention to groups who need help and advice. Teachers also find that they need to train students to recognise when the teacher is involved in a discussion with a particular group, so that they do not interrupt as soon as they come across a problem. As the students become more experienced, they realise that they can solve many of their problems without the help of the teacher by talking to other students.

Tracking Student Learning

Much can be gained by the teacher who requires each student to keep a *diary* of process notes (e.g. Rowe, 1989), questions and descriptions of achievements and problems encountered during computing. The teacher reads these reports regularly and responds. The diaries provide a valuable vehicle for keeping track of progress, and for allowing learning patterns to become visible. Simply describing a problem will often allow a student to understand the task more fully, and thus be able to solve it. Diaries provide direct access to remediation. They enable students to formalise their own thinking, to identify errors and learn from them, and to express their difficulties exactly when they are asking for help.

Student diaries of their computing efforts provide the teacher with a sense of being in charge, of knowing what is going on, and a means of keeping records of student work. They provide a means for maintaining a personal relationship with each student on a daily basis. Since the teachers are usually also still acquiring programming skills, the student diaries can become the source of a sense of comfort. They can show the teacher what he/she needs to learn, i.e. what the teacher's own homework will be.

Beyond this, the student diaries of computing provide both students and the teacher with the assurance that they are part of a collaborative learning experience. The students recognise that the teacher is working with them. Confidence about this alleviates for students and teachers feelings of insecurity or anxiety which might otherwise be present in the initial stages of teaching and learning with computers. Most educators agree that anxiety interferes with learning. As will be discussed in Chapters 6 and 7, feelings of insecurity and computer anxiety were observed even among the more experienced students in the Coombabah project.

Teachers must make sure that *back-up* copies of all the students' disks are made, to protect the students from work loss due to damage or filing mistakes, and to enable teachers to see the pattern of the work of individual students so they can plan tasks which meet their developmental needs. By examining the work on the back-up disks the teacher can also determine whether a programming problem should be solved with direct assistance by providing a tool which the student may not yet be ready to invent, or whether the student should be encouraged to persevere on his/her own.

Examination of the disks, outside school hours, allows the teacher time to work on programming problems by trying out the programs the student attempted, and trying several plans and solution paths, away from the stress of a class period. Inevitably, examination of the disks forces the teachers to think about their own next learning steps. During 1992 teachers in the Coombabah project examined back-up disks rarely, if at all. They had not set out to evaluate students' progress in computing, but had decided to restrict themselves to assessment of skill and knowledge development in subject domains. As a result of this policy both students and teachers will have missed out on valuable opportunities for learning. Assessment and evaluation are discussed further in Chapter 5.

During the actual class time the teacher walks among the students and observes, admires, comments and answers questions. The teacher can collect small groups of students around a common interest problem. Sometimes he/she will ask a student to share some work with the whole group as a teaching example or model of problem solving.

The *words* a teacher uses are important. Instead of providing a solution for a student immediately, the teacher might say: *Describe the problem. Tell me what happens. What did you want to happen? Try it now and show me*, or *Teach me what you did*. These types of response are important for several reasons. They

give both the teacher and the student time to reflect on the problem, and they require a verbal description on the part of the student. The act of describing something accurately diminishes personal emotion and allows the describer to see more clearly what has actually happened.

If, after the student has described the problem and neither teacher nor student know how to solve it, teacher and student can write a plan together in English words. Such a plan would specify exactly what the student wanted to achieve and perhaps include an example, which, if it is too difficult, can be substituted by a simpler one. Teacher and student together can write a superprocedure before they invent subprocedures, always making very sure that the first step is one the student can solve successfully.

Other tasks for the teacher include collecting, displaying and identifying resources. Many teachers use bulletin boards to stimulate the learning process. The teacher might post a weekly or daily *mystery procedure*, a new command with definition and examples of use, a challenging programming idea, a template or procedure to copy and try out, or a chart of students' names indicating their specific areas of expertise in order to make peer tutoring possible for every student. Sometimes the teacher might post an interactive program for students to copy, use, then modify and make their own.

Copying among Students

In the classroom everything students do on their computer is to some extent public. One cannot cover up a screen in the same way as some students hide their written work from one another. When learning with their own computers, it is easy and natural for students to look at, and to comment on, each other's projects, and so to learn from one another. The teacher can encourage this by giving space for wall displays of children's work and other stimulus materials. The relationships between students are different from those in a more conventional classroom. The students tend to share ideas and knowledge in an environment which does not continually stress competition.

For some teachers, the thought of students having access to each other's work in this way may raise the problem of copying. A certain amount of copying, and the right sort of copying, is perfectly healthy. Encouraging students to learn from each other is, in a sense, encouraging them to copy. Some students at Coombabah learnt about variables by copying a program directly from a wall display. Students often copy something that looks attractive, and then they make it their own by adding to it or changing it.

Too much copying is unhealthy: it tends to stop students getting involved with their own projects. As with using the printer, students can learn to be more discriminating in the ways in which they use each other's ideas and procedures. It is important that they are given the time and the guidance to do this constructively. The teacher's first instinct may be to try to impose his/her own rules about copying, but this will not necessarily achieve the result of helping the students to develop a responsible attitude. Probably the students will still copy, but they will become more devious and copy in less obvious ways.

The fact that the students are permitted to use the computers during breaks and lunch hours, and take them home, brings further benefits but increases the opportunities for copying.

Learning Contracts

This strategy requires students to enter into a contract to carry out certain learning tasks. Some of these tasks might be classroom based, others are carried out outside classroom periods at school, at home or in the community with students performing useful learning and service projects. Learning contracts allow students to become more self-directed and independent in their learning. They can also offer them the chance to explore, learn and practise real life skills in a meaningful context.

> The contract approach recognises that there are students with a wide range of both academic and personal/social abilities. By working through individual or group contracts, students are enabled to develop their abilities in relation to their own needs. Negotiation with students about the contracts and the types of activities they are going to be involved in is an important aspect of the strategy. Students need to feel responsible for their own learning. There is also an inherent motivation to work when students feel there is an element of choice. There should be negotiation between the teacher and the group, and among individual members of the group. (Queensland Education Department, 1988, p. 8)

In its publication *Practical Computer Methods: Guidelines*, the Queensland Department of Education (1988) outlines the steps to follow in presenting a contract system to students, and presents sample contracts and projects.

Fostering Problem Solving Skills

When one considers problem solving with and without a computer, it becomes evident that teachers may need to focus on similarities between programming and problem solving without computers. Similar components will have to be made explicit and practised. Those who believe in an isomorphism between programming and general problem solving ability tend to assume that the similarities in processing outweigh the dissimilarities and are sufficient to assure transferability. This assumption is yet to be tested. At present there is little historical, theoretical or empirical support for it.

Some create a one-to-one correspondence between programming and problem solving processes by noting similarities in requirements. For instance, both involve specific directions, planning, hypothesis formation, goal-oriented

behaviour, subgoal decomposition, and means-end analysis, monitoring and evaluation, so that discrepancies between what is obtained and what is intended can be eliminated. Unfortunately, there are also dissimilarities in the two processes which detract from successful transfer. For example, computer programs typically have perfect access to previous information while humans tend to lose information over time. Further, computer programs are rigidly sequenced and, once begun, continue to execute a routine to its conclusion. In contrast, human problem solvers are easily distracted by external stimuli and by ideas unrelated to the problem at hand.

Why are some students unable to cope with the problem solving skills necessary to write Logo programs needing more than three procedures? Burrowes (1985) believes that some students are generally weak in problem solving, have a poor self-image, or perhaps feel defeated by the educational establishment. When attempting to increase students' problem solving skills, one must also recognise the precursor skill of *problem posing*. Problem solving emphasises goal directed activity at the expense of exploratory behaviours. Exploratory work with Logo often leads to *problem posing*, which in turn can result in more goal directed problem solving. Sometimes students move directly into problem solving mode without having really understood what the task might require. Also, cognitive and affective variables cannot be separated. The processes of problem solving, program design and other creative work need like all academic learning to be viewed in the context of the student's motivation, interests and feelings about computing and the classroom culture.

Collaborative Learning

The term *collaborative learning* is an umbrella term which covers such activities as learning in pairs or small groups, cooperation and collaboration. Strictly speaking, the meaning of the three terms differs. *Small group* simply refers to a reduction in the size of groups or dividing the class into groups. *Cooperation* (an antonym to *competition*) means to help one another to do whatever is required for the group to succeed (e.g. Slavin, 1985, 1986). *Collaboration* refers more to the human relationships in the classroom which are expected to help students become more active, autonomous, responsible and self-directed in their learning (Whipple, 1987). Cooperation is, obviously, a prerequisite for collaboration. According to Whipple (1987), Chung (1991) and others, important characteristics of collaborative classroom activities are:

teachers and students are active participants in the learning process;
collaboration reduces the distance between teacher and students;
collaboration creates a sense of community in the classroom;
knowledge is created, not transferred;
collaboration locates knowledge in the group as well as in individuals.

Johnson & Johnson (1989) point out that the processes of interdependence, interaction and integration must be operating in the classroom if collaborative learning is to be successful. Students must see themselves as positively interdependent so that they take a personal responsibility for their contribution to the achievement of group goals; and they must engage in considerable face-to-face interaction in which they help each other, share resources, give constructive feedback and advice to one other, and be sensitive to feedback from others.

One way in which the teachers at Coombabah foster these processes is through the identification of class *experts*, who have been coached by the teacher or trained themselves in the use of particular procedures, pieces of software, etc. While this instructional strategy certainly fosters interaction and interdependence between students, it is not well received by all students. Many of them complained that the same people are selected to be *experts*, and that most students *do not have a hope* of becoming experts. These status differences might well be counterproductive for the learning climate of the class as a whole. Collaborative learning and students' becoming responsible for their own learning are fostered by having students rely more heavily on their peers than on the teacher for solving problems and evaluating outcomes. Students can be encouraged to seek help from peers and only after having done so to ask the teacher for assistance.

During collaborative learning students share, rather than compete for recognition of their efforts. They monitor and evaluate learning processes, rather than hurry to finish tasks quickly. The small group provides safe opportunities for trial and error activities as well as for asking questions and expressing opinions. In small group work more students have a chance to contribute ideas. The group also acts as a motivator and provides students with many opportunities to take on the role of *teacher* as well as learner. The system of teachers selecting and training *experts* may not be the best way of fostering truly collaborative learning.

Quite apart from the practicality in situations where there are insufficient machines for each student to have their own, the advantages of computing with one or more partners are great. Discussing ideas with others is an important aspect of any learning situation. Putting ideas into words is not just a prerequisite for such discussions but a valuable way of clarifying what and how one is thinking for oneself. The experience of starting to explain something, only to realise that one has not understood it oneself, is a common one.

Thinking aloud (e.g. Ericsson & Simon, 1984; Rowe, 1985) can be a powerful way of exposing one's own misunderstandings to oneself and within a small group. It also provides a means of sorting out the confusions by talking them through. This activity is much less threatening, and more meaningful, when the person being talked to is a fellow student, who is not expected to already understand what one is trying to explain, rather than a teacher, who might be expected to know all the answers. Sharing ideas gives the opportunity to learn from others and to see many different views of a problem.

We observed that when students in the Coombabah project were working on Logo, it was natural for them to share their views, problems and achievements with their partners. However, forming stable working relationships is not something that all children take to naturally. The literature suggests that girls may be more comfortable in collaborative work groups than many boys. The literature would suggest that, typically, the girls formed more stable partnerships while the boys tended to change partners more often and were often keen to work on their own. This was found not to be so at Coombabah. Girls did not appear to form more stable partnerships than the boys, nor did boys change their partners more frequently. It was observed, however, that more boys seemed to have a preference for working alone than girls. The class *experts* often preferred to work on their own until they had worked out a new procedure.

For the teacher to determine the partnerships among students would be missing an opportunity to encourage the students to take control. Sufficient time must be allowed for the students to choose their own partners and to change them as they desire. Also, implicitly teachers might get over the message that working with a partner is useful and important, and occasionally talk more explicitly to students who are having difficulty settling into partnerships.

Inevitably there will be some students who prefer to work on their own some of the time, particularly as they become more experienced as programmers, and it reasonable for a teacher and peers to respect this. In Coombabah we observed that often pairs of students who are experienced programmers, preferred to work on their own, but chose to be sitting near enough to one another to look at each other's screens. Although they were working on separate projects, they discussed their work and offered each other help and advice.

Intervention

A common misapprehension has grown up, especially in the Logo community, that in order to encourage students to work independently the teacher should not intervene at all. I do not agree with this view, because I regard the teacher's input as a vital component of the student's learning. The non-intervention view has resulted from a genuine concern that some teachers might play far too dominant a role in the students' work, and that the teacher's intervention can inhibit the students from developing their own understanding. A useful strategy for teachers is to stop and think before intervening, giving the students time to think out their own solution paths and giving the teacher a chance to work out an appropriate reply before jumping in. It is important to spend time just watching and listening when students are working at the keyboard, to find out exactly what they are doing, before deciding whether intervention is necessary or appropriate.

It is often tempting to try to introduce a new command or technique before students are ready for it, or to correct an error rather than suggesting how students could solve the problem for themselves. Actually, students working on their own

projects in computing often reject suggestions when they are made, only to come back to them some time later. Students learn new ideas when they are ready for them, and in a large classroom it is hardly possible for the teacher to ascertain the needs of each student at exactly the right time. If the teacher's interventions are based on the students' own progress, but without any pressure for ideas to be taken up immediately, then students can make the ideas suggested to them by the teacher their own when they are ready to do so. It is impossible to make hard and fast rules, but the suggestion of never touching the student's keyboard is a good starting point. Obviously an exception to this would be if there is something wrong with the disk and the teacher wishes to get the student back to work quickly. Even in such a situation the ideal response would be to help the student fix the problem herself/himself by verbal suggestions which the student might try on the keyboard.

Holding back from intervention to allow students time to find their own solutions, to develop their own projects and to control their own pace of work is often an important part of the teacher's role. It takes time for students to become committed to the work they are doing. At the start of a new project students might work at a relatively low level before they are ready to tackle the more difficult aspects. Students with their own laptops and Logo projects are able to learn in a natural way because they have control over what they are working on, and because they can control *when* they tackle difficult ideas.

Mistakes. Another way in which students can take control of their own work in computing is in judging their own success or lack of success. Because students have set their own goals, and usually have instant feedback on their actions from the computer, the need for the teacher to say that something is *correct* or *wrong* is much reduced. In fact, a teacher might well be embarrassed when he/she says *That is nice*, commenting on an attractive display on the screen, and is told by the student *No, this isn't what I wanted to do at all.* Students who are used to having their work *marked* by teachers may take some time to get over regarding errors as wrong and something to feel embarrassed about. They will need encouragement from their teachers to change this feeling. Promoting debugging as a respectable, useful and at least mildly enjoyable activity is one way of developing a positive attitude to making and correcting mistakes. Using the term *bugs* instead of mistakes is likely to help overcome existing prejudices.

An important feature of Logo and other software is the error messages which appear on the screen if a command is typed which the computer cannot interpret. Unfortunately, these messages are often not read by novices, or they are found difficult to interpret. In the early stages of computer use most error messages tend to reflect typing errors. Learning to read and interpret error messages is an important step for students in learning to program and in taking responsibility for their own work.

Asking questions. It is often easier to make interventions in a relaxed way by asking questions. When students are involved in their own projects, it is clear that they themselves know what they are aiming at and that the teacher is on less

familiar ground. This is the reverse of the traditional classroom situation and can lead to noticeable changes in the dialogue between teachers and students.

In traditional classrooms, the purpose of almost all the teachers' questions is not to get information but to test what students know. Questioning becomes ritualised by both teachers and students and loses its potential as a teaching technique, or even as a normal means of interaction between people. In computing environments the teacher is more often genuinely asking for information (e.g. *How did you draw that part? Which procedure draws the eyes? How will you get the turtle back to the right place?*) and the students realise this. Questioning becomes the basis of a conversation rather than an interrogation. Asking appropriate questions can be a powerful way in which the teacher encourages students to explore extensions to their projects, and to introduce new challenges.

TEACHING TO FACILITATE INDEPENDENCE

Teachers are able to touch students in many ways. They implement educational policy and curriculum content. Even more importantly, they establish the educational climate, and structure learning experiences. They have almost complete power over the processes which take place in the classroom, and in the final count these processes contribute more to education than does drill and practice. One of the most important goals for the teacher is that of encouraging students to become responsible for their own learning.

Teaching and learning with computers provides the potential for natural human relationships to develop between teacher and students as they collaborate to solve a problem that arises in one of the students' projects. As noted above, the role of the teacher becomes more like that of a collaborator, and their authority is based on their knowledge and ability to help, rather than on personal and professional status. When the teacher is seen as cooperating in students' activities, rather than judging them, the relationship between students and teacher can begin to approach the ideal described by Jerome Bruner in *Toward a Theory of Instruction*:

> I would like to suggest that what the teacher must be, to be an effective competence model, is a day-to-day working model with whom to interact. It is not so much that the teacher provides a model to *imitate*. Rather, it is that the teacher can become part of the student's internal dialogue -- somebody whose respect he wants, someone whose standards he wishes to make his own. (Bruner, 1966, p. 124)

One factor which contributes to this more natural relationship is that the teacher is often talking to students individually, or in small groups, rather than to the class as a whole. This obviously requires a different manner and tone of voice, and it is made easier if the teacher can sit with the students rather than standing over them. Of course this could lead to problems if the teacher's attention is

completely taken up with one group, to the exclusion of the rest of the class. Simple strategies such teachers sitting in such a position that they do not have their back to the rest of the class when talking to a group can help to maintain contact with all students.

What is being said in the classroom and how it is being said, and what students and teachers do in the classroom, greatly affects learning. Certain teacher behaviours have a particularly strong and direct influence not only on student learning and educational achievements, but also on motivation, self-concept, social relationships and how students think about learning itself in and beyond school.

Many of the teacher behaviours which have been shown to invite, enhance and maintain high levels of communication among students can be seen as falling into one or more of the four following categories:

1 *Structuring the classroom* flexibly to allow for individual, small group and total group interaction as may be required. Managing the resources of time, space, materials, energy, interest and motivation to facilitate participation in the communicative process by all students. Making *thinking for oneself* and *taking charge of one's own learning* an important and acceptable educational objective for each student.
2 *Questioning* to help students acquire new and access previously acquired information and experiences (input), attaching that information to previously acquired knowledge and restructuring and transforming it into meaningful relationships (processing), and applying what has been acquired, restructured or processed in other ways, in a variety of different, including novel situations (output).
3 *Responding* to make students aware of their ability to think and learn independently, helping them extend the power of this independence and, above all, maintain their interest in continuing to acquire more effective ways of learning.
4 *Modelling* of the types of behaviours you wish your students to acquire. As part of the day-to-day activities in the classroom, speak about and demonstrate the strategies you yourself use as you go about cognitive tasks, and encourage class discussion about the strategies different people are finding effective (or not effective).

Structuring the Classroom

Structuring the classroom refers to ways in which the physical environment of learning, environmental resources such as time and materials, and human resources such as energy, interest, motivation, etc., are used. Every classroom is structured in one way or another, either consciously or unconsciously, either

directly or indirectly. Even an *unstructured* classroom imposes a structure to which and within which students and teachers react and interact.

Structuring the classroom for high levels of communication should be conscious, deliberate and clearly based on the desired objectives for the students. Having planned in advance which learning tasks are to be accomplished and what types of interaction are to be achieved, the teacher may wish to state some ground rules, describe and explain the objectives, place limits and constraints, and create an organisational pattern which he/she expects might best accomplish the desired objectives. Research has shown that allowing students to work cooperatively promotes more independent thinking and higher reasoning strategies than do more competitive and individualistic learning situations. In fact, most successful programs designed to develop higher order thinking skills prescribe cooperative learning activities. It was found that the social setting provides occasions for modelling and practice. Skilled thinkers (often the teacher, but sometimes more advanced students) can demonstrate desirable ways of attacking problems, analysing text, or constructing arguments. Students can scaffold complicated performances for each other. Each one does part of the task, and, by working cooperatively, students can arrive at solutions that one student could not manage alone at that particular stage. In addition, mutual criticism during shared work provides the feedback that can help refine or restructure individuals' knowledge and skill.

Teachers who encourage and promote independent thinking and learning tend to provide a classroom climate where:

- the students are able to see themselves as being in the decision making role,
- the students decide on the strategies they will use to accomplish given tasks,
- the students determine the correctness or incorrectness of an answer based on data they themselves are producing and are able to validate,
- the students are involved in setting their own goals and means of assessing the accomplishment of those goals.

Also, the reward system in such classrooms is intrinsic rather than extrinsic (i.e. derived from internal motivation to learn, an intellectual curiosity about phenomena, a striving for competency and accuracy, a sense of responsibility to be a productive member of a community of learners, and a desire to emulate significant respected others).

Questions and Answers

Aspects of teacher questioning which have been found to have significant effects on initiating real communication in class include the types and levels of questions asked, teacher wait-time (i.e. the pause between the end of a teacher's question and the beginning of a student's response, or after a student's response and teacher feedback), and teacher follow-up to student answers.

Dillon (1984) made the distinction between two types of classroom interaction to which he refers as *recitation* and *discussion*. Recitation is totally teacher-centred and characterised by recurring sequences of teacher questions and student answers, in which students *recite* what they have learnt previously or what they are currently learning in response to the teacher's questioning. Real classroom communication, however, involves group interaction in which students *discuss* what they know as well as what they do not know or understand. More than one point of view is brought forth and considered. The teacher acts as a facilitator by creating a non-threatening atmosphere of equality for students, by providing clarification and guidance, by unobtrusively moving the discussion into desired directions. Capable adept use of discussion is a most important tool for the facilitation of independent thinking and learning.

Students who are working on their own projects can solve many of the problems themselves or through discussion with peers, but sometimes they will demand help from the teacher. A teacher who is less familiar with working with, for example, Logo may feel unsure about the best way to respond to these requests, because there is a tension between wanting to help the students get on with the job, and wanting them to think about solutions themselves. The most appropriate strategy in any particular case will naturally depend on the circumstances. The following examples illustrate this for some typical questions:

How can one move the turtle without drawing lines? This type of question is straightforward. The student knows exactly what he/she requires. The knowledge sought is not linked to any conceptual understanding, and there is no way the students could work it out for themselves. There is a Logo command to lift the turtle's pen, and the teacher's response must be to provide this information or to direct the students to a resource, e.g. the handbook, from which they can find out.

How do we draw a circle? This request is less straightforward and there are several possible levels of response. Because the teacher's aim is to encourage problem solving and independent learning, he/she would regard this problem as one the students could work out for themselves and encourage them to do so. Pretending to be the turtle would help students to understand the need for short forward movements, alternating with small turns. This involves either implicitly or explicitly refusing to answer the question directly. How a teacher does this will depend on the students involved, and the teacher's personal style.

How does REPEAT work? Here the student knows the purpose of *REPEAT*, but needs to be reminded of its syntactic form. Again, the most appropriate response depends on the student and the circumstances in which the question is posed. A straightforward answer may suffice, as most students will not need much encouragement to explore commands of the form *REPEAT 100[FD 100*

RT 155]. Any activity of this nature will provide plenty of practice with the command and will allow students to discover a number of mathematical relationships at the same time.

Deflecting questions which students should answer for themselves can have positive effects in terms of encouraging independent thinking, but it can also be disconcerting for the students. To them, asking how to draw a circle may not seem any different from asking how to raise the pen, and the teacher's behaviour in answering one question directly but not the other may appear inconsistent and/or obstructive. Some students' answer to the suggestion of walking like a turtle etc., when they have asked for help in drawing circles, might be to make the wheels a different shape. There are countless questions students ask when their work is important to them. The teacher's response may be to give them information, or to create a situation where the students can discover a solution for themselves. What is important is that the students feel free to ask questions.

Positive and Negative Responses

The way in which the teacher responds to and interacts with students in the classroom determines the degree of trust, warmth, openness, rapport and psychological safety in the classroom. It also strongly influences the preparedness of students (especially the more reticent ones) to take risks in trying out new ideas and computing strategies.

Some teacher responses such as criticism (and other ways in which students are *put down*) and praise result in the termination or temporary closing down of communication with the student. Other responses result in extending or opening the communication process. Examples in the latter category are acceptance, the use of silence, clarification and facilitation.

Criticism This is the expression of a negative value judgment. There is an abundance of research evidence to show that criticism does not promote cognitive or affective learning, and that it lowers students' self-esteem and achievement. When a teacher reacts to a student's performance with brief, negative words such as *poor* or *wrong*, he/she is likely to terminate interaction with the student as well as the thinking process of the student. More subtle and less negative signals of the inadequacy of a response might be *You are almost right, can anyone add to this answer?* or *You are getting close*. Ridicule, sarcasm and other responses which are designed to put the student down should be avoided at all cost. Criticising students and making them feel a failure certainly does not enhance thinking and learning.

Responses which are more useful than criticism in promoting student thinking and learning tend to be the ones which represent extending and opening behaviours. Prominent amongst these are the appropriate and not too frequent use

of praise, large amounts of acceptance, clarification, facilitation and the skilled use of silence.

Praise Praise might be seen as the opposite of criticism. It manifests itself in the expression of positive value judgments such as *good, excellent, very useful, great*, etc. While most educators strongly advocate the use of praise to reinforce desired behaviours and for the building of self-esteem in students, there are some problems related to the indiscriminate use of praise. Praise builds conformity and it is thus less useful where our goal is diversity. The effect of praise on some students is to make them dependent on others for their feelings of self-worth rather than on themselves. Also, praise, like criticism, tends to terminate the interaction between student and teacher. It is important for teachers to recognise this, and use praise judiciously with those students and in relation to those objectives for which it is suitable (e.g. with reluctant and unmotivated students, young children in rote learning of low level cognitive tasks).

The teacher's long-term goal should be to decrease the use of terminal behaviours including praise and criticism. Teachers can replace their habit of offering praise too frequently with an enlarged repertoire of response behaviours which have been shown to be more conducive to developing student thinking, learning and self-esteem. Amongst these are acceptance of a variety of views, clarification, facilitation and silence.

Acceptance This term describes responses which are non-judgmental and non-evaluative. Neither words nor gesture, posture, etc., give clues as to whether the teacher regards the student's response or idea as *correct, good, bad, worse, better*, etc. Alternative ways of reacting to a student's answer are by acknowledging it, paraphrasing or summarising it, applying it or comparing it with another idea.

The intention of acceptance is to build a psychologically safe classroom climate in which students can take risks, feel that they are entrusted with the responsibility of making decisions and can explore the consequences of their own actions. An atmosphere of acceptance encourages students to examine and compare their own views, feelings, reactions, values and criteria of success with those of other students as well as with those of the teacher. Even when students' views, feelings, etc. differ from those of the teacher in a seemingly unacceptable way, the teacher can still accept them temporarily, because he/she realises that only the student is able to modify them. The task of the teacher is to provide information and to guide the discussion in the classroom in such a way that it facilitates the processes which lead to students' modifying their feelings, criteria for action, etc. making them more consistent with reality and the demands of the task or the situation.

The classroom where independent thinking and learning are fostered is one where a spirit of inquiry prevails. Student questions and intellectual challenges are valued. The teacher admits uncertainty: *We are not really sure how evolution*

comes about, I am not sure about my interpretation of this poem, I continue to find new things in it. In this way the teacher emphasises education as an exploration of the unknown, as well as learning what is known.

Teacher acceptance encourages *problem finding* on the part of students. In many classrooms quick answers and solutions are sought, encouraged and valued. In an independence oriented classroom, students are taught and encouraged to identify problems, to wonder and to speculate. The unthinking person may observe graffiti and either smile or frown. The thinking person wonders why in Europe graffiti is so often political while in the USA it is more commonly scatological and in Australia childish. The teacher nurtures a problem finding disposition by encouraging students to ask questions of their own, not just answering the questions posed by others. *Here are some data about income distribution in Australia, what questions could we ask? We'll be looking at the role of the nuclear family in Aboriginal communities, what questions would you like to have answered?* Note that acceptance can be demonstrated in different ways, e.g. it can be quite passive, active or even empathic.

Passive acceptance refers to instances in which the teacher merely receives and acknowledges what the student says, without making any value judgment. It shows the student that his/her response has been heard. Examples of passive acceptance behaviours teachers can use are: *That is one possibility, I understand, Could be, Hmm.* Non-verbal passive acceptance behaviours include nodding of the head or writing the student's statement on the blackboard.

Active acceptance refers to instances in which the teacher demonstrates an understanding of the student's response. The teacher actively accepts by reflecting (not merely repeating), extending, building on, comparing or giving an example based on the student's response. While rewording the student's response, the teacher strives to maintain the intent and accurate meaning of the student's idea. Active acceptance is stronger than passive acceptance because the teacher not only acknowledges that the student's message has been received, but also that the intent of the message has been understood.

Empathic acceptance is an acceptance of feelings as well as the products of thought. It means that the teacher not only hears the student's ideas but is also sensitive to the emotions underlying or accompanying these. Teachers can show empathy when they express similar feelings to those of the student from their own experience. Empathic acceptance does not mean that the teacher condones acts of aggression or destructive behaviour. Rather, it demonstrates an understanding and acceptance of the emotions that produce such behaviours.

Clarification This is similar to active acceptance in that both behaviours are concerned with the teacher's understanding of the student's idea. While active acceptance conveys that the teacher understands, questions of clarification convey that the teacher is seeking understanding but requires more information. Nearly 30 years ago, Flanders showed that student achievement is higher in classes where teachers use, build on, extend or clarify students' responses. When

teachers encourage students to elaborate on their answers and use other methods of clarification, students tend to increase the consistency of their thinking, i.e. they become more task oriented and purposeful in their problem solving and learning.

One of the most compelling reasons why teachers should make frequent use of clarification is that it contributes to the development of students' *metacognitive skills*. There is a high correlation between the degree of metacognitive awareness and the level of performance on complex problem solving tasks. Students become better problem solvers and learners if they are able to become aware of and talk about the strategies and steps they use in their problem solving and learning.

Some students follow computing instructions and perform tasks without asking themselves why they are doing what they are doing. They seldomly, if ever, evaluate their own learning strategies or the efficiency of their own performance. They have virtually no idea what they are doing when they perform a task, and are thus unable to explain the strategies and steps they used. When the teacher asks students to explain their work, i.e. to show how they arrived at a solution or to share their rationale for a certain procedure, the teacher causes the student to use metacognition. For 40 years research evidence has been building up for the view that thinking and talking about thinking leads to more thinking. Causing students to talk about their thinking and learning processes during and after performance enhances their ability to think.

Facilitation Facilitating the acquisition of information, knowledge and skills is a basic aim of teaching. To do this the teacher must be sensitive to and able to perceive students' needs, provide information and make it possible for students to do so themselves. Knowledge of results, i.e. feedback, is the most important variable governing the acquisition of skill, but also in the development of independent learning dispositions and habits of thinking. Note that there is a difference between rewards and feedback. Rewards can either control behaviour or give information about competence. If students perceive the teacher's praise as controlling, their intrinsic motivation is likely to decrease. If, however, students perceive rewards as providing feedback about their skill or competence, intrinsic motivation is likely to increase.

Silence Silence and waiting time are important aspects of teacher/student interaction. Many classrooms could do with a more deliberate pace rather than encouraging impulsiveness. The teacher asks a question, expects an immediate answer and calls on the first student who puts up his/her hand. Such rapid-fire recitations can be useful in several ways. They facilitate the assessment of a single student's knowledge, permit rehearsal of facts and keep students attentive. However, if the aim is to develop problem solving and thinking, as when learning with computers, this style of interaction is counterproductive. Students need time

to deliberate, i.e. to reflect about alternative possibilities, to weigh the evidence and to come to a tentative conclusion.

Research has shown differences in student behaviour in traditional classrooms where teachers waited after asking a question or after a student responded. Teachers who wait just a short time, i.e. one or two seconds, tend to receive from their students short, often one word, responses. Teachers who wait for longer periods tend to elicit responses based on more complete thought, and in whole sentences. Also, there is an increase in the creativity of responses as shown by the more frequent use of descriptive and modifying words, and an increased speculativeness in students' thinking. Interactions among students in the group are increased, the number of questions students ask increases and, most important of all, reticent, shy and slower thinking students begin to contribute. The same processes are likely to operate in the classroom where students learn with computers.

Teachers can communicate their expectations to students through the use of silence. Teachers who ask questions and then wait before they invite a student to answer show that they expect an answer but also that they have faith in the student's ability to answer, given sufficient time. Teachers who ask a question of a student, wait only a short time, and then give the answer, call on another student or give a hint, only demonstrate to the student their belief that he/she is really unable to answer the question, i.e. is considered too poor a student to offer an answer or to reason independently.

Modelling

How do students choose their models and how do they know what to emulate? Teachers and some class *experts* provide immediate models. In addition, however, students need to be presented with models and case studies of successful *experts* in computing and other domains. Students can learn a great deal through the direct observation and the study of detailed accounts of peers and adults (including their teachers) struggling with problems. This allows them to observe, discuss and understand why certain processes are operating differently in different situations and with respect to different tasks. They will learn that creative thinking and independent learning habits are not limited to a given age group, to certain occupations/professions, to particular ethnic groups or social classes, or to scientific and other scholarly effort. They need to observe that the best thinkers, including their teachers, more experienced students, parents, and significant others can be wrong, and that the path to success is often uncertain and may be full of torture.

Modelling by the teacher serves to share with students not only what the teacher might be thinking about the content to be learnt, but perhaps more importantly, information and feelings about the processes of learning and problem solving.

CONCLUSION

The implications of teaching to facilitate independent thinking, problem solving and self-regulated learning, with and without computers, include the enhancement of communication processes between peers, and between students and teachers, an enrichment of teacher conceptions of individual differences among students in motivation, interest, learning style and information processing, previous experience and behaviours, an improvement of instructional methods, and the broadening of educational goals and outcomes. This type of teaching leads to new kinds of educational structuring which extend beyond the boundaries of the classroom and school. The teaching and learning advocated in this chapter is characterised by shared knowledge among teachers and learners, students and teachers sharing planning and control with respect to the content and processes of learning, and students accepting personal responsibility for their learning as members of a community of inquiry and learning.

5

Assessment and Evaluation

ASSESSMENT VERSUS EVALUATION

In educational contexts, assessment, as an aspect of evaluation, if not a prerequisite for it, involves gathering and transmitting information which is relevant to and assists in making certain kinds of decisions.

The *Encyclopaedia of Educational Evaluation* (Anderson, Ball, Murphy & Associates, 1975) defines assessment as a process for gathering information which can meet a variety of evaluation needs. The process of assessment involves multiple indicators and sources of evidence, and in this sense is different from testing.

> Assessment, as opposed to simple one dimensional measurement, is frequently described as multitrait-multimethod; that is, it focuses upon a number of variables judged to be important and utilizes a number of techniques to assay them . . . Its techniques may also be multisource . . . and/or multijudge. (Anderson et al., 1975, p. 27)

Assessment is the process of collecting and organising information or data in ways that make it possible to judge or evaluate performance, the operation of a program, etc. Assessment data of any kind are no more than indicators of a phenomenon. The evidence associated with such indicators must be unambiguous to the extent that the context and means of its collection are understood by the students, parents, other teachers and whoever else might wish to use it. Assessment data provide evidence, but the evaluation of that evidence can still be open to interpretation as different people might form somewhat different judgments concerning the implications of the data.

> It therefore seems appropriate . . . to limit the term assessment to the process of gathering data and fashioning them into an interpretable form; judgments can then be made . . . Assessment, then, as we define it, precedes the final decision making stage in evaluation. (Anderson et al., 1975, p. 27)

Hayman, Rayder, Stenner & Madey (1979) suggest that for assessment to be used effectively, four criteria must be met:

1 The information resulting from the assessment must reduce uncertainty. If uncertainty is not reduced, no information is provided by the assessment.
2 A proper format must be chosen for representing, transmitting, receiving and relating assessment information.
3 The recipient must understand the meaning of the information. In other words, the information must be accessible to the intended user.
4 The information must be capable of motivating human action.

IMPORTANT ISSUES

The goal of introducing computers into the classroom is to assist with the intellectual development of students. Assessment and evaluation of student progress is an integral part of this process of development. Evaluation is needed not only to judge the intellectual and personal development of the students, but also to determine the effectiveness and validity of curriculum content, classroom organisation and the teaching strategies used.

How can learning processes and educational outcomes in computing be evaluated? At present, there appears to be some disagreement and uncertainty about what is to be expected from students who are learning with computers. This uncertainty is the reason for the current lack of agreed curriculum guidelines and criteria for student assessment. Individual teachers thus need to clarify for themselves what they expect to achieve through the use of a computer and the computer-based activities in the classroom. In learning with computers, as in all other areas of education, assessment is a positive aid to learning and instruction. The important components which need to be evaluated fall into six broad categories: computing skills, knowledge, awareness, attitude, learning processes and learning outcomes.

Assessment should have both formative and summative aspects. The process skills used by students to complete tasks are at least as important as the final product of an assignment. Learning outcomes may show whether a student has acquired certain facts, rules and procedures. However, measures of learning outcomes do not provide any information about the way in which the assessed knowledge and skills have been acquired, nor do they tell us what prevented a certain individual from developing the required knowledge or skills. The major purpose of formative assessment is to provide diagnostic and remedial feedback to the student; in addition it will provide information which will update the student's profile of skill and knowledge development for the teacher. Assessment should be concerned with the qualitative aspects of the learning process as well as the final outcome.

Some aspects of computer awareness can be assessed through written or oral procedures, even by means of multiple choice tests. Student attitudes to computing can be assessed by talking with individuals or groups of students and through questionnaires. However, the physical handling of the hardware and

software, and computer programming, are best evaluated through the observation of performance and the evaluation of the workability, efficiency and effectiveness of the final outcome or product. The most revealing way for students to demonstrate their abilities and skills is through practical application. This encourages them to display their knowledge and skills in the planning, design, creation and appraisal of solutions and other cognitive outcomes. This also allows the observation of student attitudes, learning styles and other personal characteristics, their ability to concentrate, persevere and apply themselves to tasks, their capacity to work with others and on their own.

In classrooms where students are working with computers, we are likely to see a lot of group activity and discussion. Students move about the room to watch other students, to ask for assistance, to debate and argue their view. Students gain valuable insights through listening to the comments of others about their work and ways in which it could be improved. Pieces of work at various stages of completion are evident in the classroom. Because students may work on individual or group projects for hours, days or weeks, they are engaged in a variety of activities as they experiment, develop ideas, test new theories, plan and draft solutions, revise and refine their work. Through these activities, students show their understanding, difficulties, and their developing knowledge and skills. Self-evaluation (written or oral) is an excellent method of enabling students to reflect on their learning.

As with other areas of learning, how computing knowledge and skills develop should be monitored regularly and as an integral part of classroom learning and teaching. Much information can be gained quite quickly by the teacher by simply talking to the students about their work, what they have learnt, enjoyed or disliked. At this time teachers would also discuss with the student how they think the student is working, decide whether quality and quantity of work is of an agreed standard (and if not, to try and find a solution with the student to overcome the problem). Assessment which occurs within the context of learning has a certain naturalness about it. Realistic competence is displayed by the students as members of a community of learners, in contrast to the decontextualised, isolated, and competitive character of testing as it is often carried out. As members of groups and the broader classroom community, students learn to assess their own performance and development, and they come to rely on group feedback and peer commentary about their ideas and work strategies (in addition to feedback from the teacher) as forms of assessment. They learn to understand the importance of the part individuals play in group work, and they learn to evaluate their own contribution and that of others. In these contexts, students are able to monitor their performances and observe the performances of more competent as well as less competent peers more consistently than is possible in situations where learning and problem solving proceed individually and silently, and where all that matters is the answer or end product.

As noted in the previous chapter, students can be encouraged to keep a personal and/or group record of their daily progress. In a special computer diary

file they can describe what they did, as well as noting comments and thoughts about some of the decisions and choices they made while writing a procedure, or during the creation of a story. Sometimes students might be given an opportunity to write a small reflective piece in which they review their past work and might begin to develop a sense of their growth and learning over time. The latter is an important dimension of formative assessment.

As noted above, assessment and evaluation need to be formative and summative, but above all they need to be continuous. A balanced approach to the evaluation of learning in all areas of the curriculum requires the collection of many kinds of evidence over a long period of time, i.e. the whole term or school year. Assessment should identify strengths and weaknesses, and point to how learning processes and learning outcomes of students can be improved (cf. criterion 4 noted above). Assessment includes describing and monitoring student progress, and making summary statements of achievements in terms of both learning processes and outcomes. Both of these should be based on previously determined goals and on specified work requirements. Judgments of capabilities could then be guided by the outcomes described for each band of schooling (e.g. lower primary, upper primary, lower secondary and upper secondary).

Over the past five years, most State Education Departments in Australia have produced charts listing goals for attainment in educational computing. At present, there is little communality between these charts and no followup as to the usefulness of the goals themselves is available. Having declared technology education as one of the eight mainstream areas of learning in the Hobart Declaration in April 1989, the Australian Education Council has mounted a major National Technology Education Project. The resulting *National Statement on Technology Education for Australian Schools* is expected to include detailed statements of goals and expected achievements. The document should be released in the near future. In the meantime, the following generally agreed upon broad aims might serve as initial expected outcomes for primary and lower secondary school students.

EXPECTED OUTCOMES

Two quite basic outcomes of educational computing can be expected in primary and lower secondary school classrooms:

1 students will come to feel confident and comfortable about using the computer as a learning tool; and
2 they will use this tool regularly across the curriculum to achieve learning objectives and to solve problems in the context of their daily classroom activities.

Indicators which could serve as evaluative measures for (1), i.e. to assess the extent to which students are confident and comfortable in using computers as tools, might be as follows:

1-0 student does not use the computer;
1-1 student uses computer only when directed and for limited purposes, e.g. drill and practice;
1-2 student uses computer regularly with guidance from the teacher;
1-3 student uses computer in group situation, interacting productively;
1-4 student confidently uses software or procedures written by others in various domains and for various purposes;
1-5 student is able to match a particular software application to his/her specific need;
1-5 student takes initiative to use computer appropriately;
1-6 student uses the computer for his/her own explorations;
1-6 student explores computer use for for divergent purposes.

Indicators for (2), i.e. to assess the extent to which students regularly use computers to solve problems across the curriculum, might be:

2-0 no problem solving use of the computer;
2-1 attempts to use computer for problem solving in one or more areas of the curriculum;
2-2 computer used regularly for problem solving activities in a particular area of the curriculum;
2-3 student displays an ability to select a software application or module which matches his/her problem;
2-4 student uses subject specific problem solving procedures in mathematics, science, social studies, language, etc., and is able to help peers with this;
2-5 student tries to use computer for problem solving across the curriculum, e.g. attempts integrated use of applications such as word processing, graphics design, databases, writing of procedures, information processing (i.e. retrieval, analysis and presentation);
2-6 student uses computer across the curriculum to process information by organising, manipulating, analysing and synthesising information, and is able to help peers.

MONITORING PRODUCT AND PROCESS

In addition to the above discussed broad outcomes of learning with computers, there are more difficult issues of assessment and evaluation. Key questions include:

1 How can we assess or evaluate the outcomes of work in educational computing? What are the criteria for assessment?
2 How can we assess the processes of thinking and problem solving taking place during computing, rather than facts and final solutions?
3 What is the final solution if learners are working with programs and in media which invite them to return and encourage further editing?
4 How do we assess the work of individuals when the work is collaborative?

With respect to questions 1 and 2, it is obvious that all learning and teaching must include monitoring. Further curriculum preparation and planning depends on the results of the ongoing monitoring of what has been covered. The techniques outlined later in this chapter will be applicable for courses in computing as a subject as well as for the evaluation of the development of computing knowledge and skills in integrated courses.

There are many factors to be considered in monitoring both a course in computing studies and the use of computers as a tool integrated in subject areas. In both cases monitoring helps ensure that the requirements of the syllabus are met. Objectives, perspectives, content, approaches to planning units of work and different teaching techniques must all be considered.

In the absence of a formal curriculum statement and syllabus, teachers will be setting their own goals and devising suitable learning sequences for their students. These will relate to the broad areas of computer awareness and attitudes to computing, knowledge about computers and computing, and computing skills. Whether computing is taught as a separate subject or is integrated into other subject areas, the processes involved in learning computing and in learning by means of computing include the following:

analysis of the topic or task,
comprehension of what is required,
finding out facts, rules, etc., which might apply,
planning,
preparation and presentation of information,
use of software and hardware,
design of solution,
execution of task,
self-monitoring and evaluation,
sensitivity to feedback.

The problem of monitoring is not so much a matter of evaluating what is done but rather with finding methods of keeping track of what is happening as learning and teaching evolve. Whether a curriculum is written down in its entirety before teaching any part of it, or developed as teaching proceeds, there must be careful documentation and monitoring of all the parts.

Different teachers will have different strategies to monitor progress throughout their teaching. Some will prefer to have pages in a book, or computer records that allow portability, others wall charts of various sizes so that the whole course can be seen at a glance. This is a matter for individual preference. The only necessity is that there is monitoring. A later section of this chapter entitled *Evaluating Progress* provides specific suggestions as to how such monitoring could be accomplished.

With respect to questions 3 and 4, concern has been expressed (e.g. Heppell, 1989) that we have not yet fully appreciated the significance of the non-linearity of working with computers. In particular, it could be argued that we have not yet developed a language for the use of such non-linearity. In our attempts to evaluate the work of students who use computers we behave as though we are evaluating the outcomes of working with pen and paper technology. We look for *originality*, for *first drafts*, and for work that has been *completed* or preplanned. Unfortunately these terms are less useful in the computing environment, because they prevent us from actually making use of the freedom provided by the medium.

The facility to return to a piece of work, to re-use, build upon and modify previously produced material is a major benefit of the computer. It is not just labour saving in the conventional sense of time and physical effort, but it allows all learners to build upon a store of personal as well as collective past experiences and ideas. This is an ability that experienced learners have in abundance. The reworking of ideas and knowledge is an essential part of Bruner's (1966) *spiral curriculum*. Whereas linear curriculum models forge limited links between pieces of knowledge, engendering perceptions of completion and closure, in the spiral curriculum knowledge has multiple links and is constantly growing in an associative network and is therefore more readily available to use.

Should we reward students who use all the search, sort and graphical tools at their disposal, or those who ask good questions and achieve relevant and useful answers? Even here we are concerned with summative evaluation, the end-product of the learning experience, rather than formative evaluation of the process of learning. A Logo case study described by Hoyles, Sutherland & Evans (1986) shows how we can be misled by an evaluation which is restricted to the end-product. It reports the work of pairs of children over an extended period of time. While most pairs in the class had both successes and setbacks, two boys continuously produced exciting visual patterns using recursive procedures. Further analysis showed, however, that the complex patterns of one week varied little in structure from those of previous weeks, even though the patterns were often visually dramatically different. The boys were manipulating variables in a procedure by *handle turning*. What is even more disturbing is that they had borrowed the initial procedure that was being manipulated from a neighbouring group in the first week of the project. Summative evaluation of their work marked them as being highly successful, but more careful evaluation of the processes they were using revealed that the two students were *stuck* in their own

procedural loop, trapped by initial success and now unwilling or unable to experiment.

Heppell (1989) argued for a technological solution to assessment in which the computer would keep a log of the investigative strategies of the learners. What is suggested is not a skills *tick list* so much as a monitoring of an ongoing process. This method of monitoring could go a long way towards forcing students and teachers to focus on process, although the logged data would still need very time-consuming evaluation. In some instances the computer can be set up to manage simple instruction, e.g. to keep track of students' performances on drill and practice activities. Other computers are set up for diagnostic testing. The hoped-for outcome of these measures is to free the teacher for more essential work.

Computer-based activity of the more open-ended variety can provide teachers with new insights into what their students can do. Anecdotal accounts have for a long time described how teachers have learnt new things about their students' capabilities as a result of observing them interacting with the computer and peers (e.g. Burns, Cook, Dubitsky, 1982; Papert, Watt, diSessa & Weir, 1979).

With a greater emphasis on skills of abstraction and comprehension, what student achievement consists of and how it is measured will need to change (Fredericksen, 1984). For example, the advent of the pocket calculator has meant that mathematical operations and estimation can be emphasised over calculation. Word processors have resulted in a new emphasis on the writing process, as opposed to spelling and the forming of letters. Programming will allow the observation of students' planning, monitoring, problem solving and decision making skills.

Determining whether a student is a good problem solver who can imagine multiple solutions, plan solution strategies, and estimate outcomes is very different from counting how many problems a student can answer correctly in a given time. A composition may no longer be judged simply by the number of spelling and grammatical errors it contains.

Intelligent computer systems, which are presently being developed, will make it possible in the future to promote and diagnose student performances in new ways. Based on the student's performance, these systems might prompt the student to reconsider an answer, demonstrate a different process for solving a particular problem, or ask the student to indicate why he/she thought a particular response was correct. Other types of intelligent systems might help teachers understand how students learn and solve problem by analysing students' errors (e.g. Burton, 1981; Ohlsson, 1986).

GROUP WORK

Learning with computers usually involves a considerable amount of group work. Formative as well as summative evaluation of group work is important. Assessment at various stages of a group project provides the teacher with more

opportunity to observe the contribution of individuals to the group effort than assessment of the final product can.

In the assessment of performance on tasks tackled by collaborating students the same mark might be given to each of the students in the group. Where this is unsatisfactory, the contribution and achievement of individuals to the group effort could be estimated as well. For example, a common mark might be given for the product of the group's work, but in addition individual members of the group might be awarded marks above or below this group mark, according to the level of their contribution. The assessment of the contribution of individuals is judged on the basis of observation by the teacher, group discussion and subsequent peer assessment, or a combination of both. Our research in the SUNRISE classes at Coombabah showed peer assessment to be objective and highly consistent with teachers' judgments. The latter observation is supported by the literature relating to peer assessment.

Some criteria for the assessment of the processes of group work might be provided by answers to the following questions:

Did everyone contribute?
Was there a group leader?
Did *experts* in different aspects of the task emerge?
Did everyone have an opportunity to contribute?
Did everyone choose to contribute?
Did students take turns using the keyboard?
Did the students select one keyboard operator?
Was there evidence of group planning and decision making?
Did the students vote on how to proceed?
Did some students lose interest or show frustration?
How did the students evaluate their work?

It is also possible to focus on the performance of individual students within groups:

Did Student X contribute to the whole group's activity or task?
Did Student X show that he/she is able to listen to others as well as talk?
Did the student share in the *more difficult* as well as in the easy aspects of the task?
Did the student show self-motivation?
Did the student accept responsibility for his/her own suggestions, actions, etc.?
Does the student perceive the project to be his/hers?

DIFFERENT ASSESSMENT PROCEDURES

Standardised tests of computing are not currently available. Even if such tests became available, they would be of limited value for the assessment of process skills. More suitable alternative methods of assessment include direct observation, screening procedures, informal interviews, structured interviews, situational try-outs, work samples and analyses of tasks and learning contexts. Each has advantages and disadvantages, and requires differential time commitments and varying levels of expertise on the part of the teacher utilising the method.

Direct observation can be spontaneous or systematic. During spontaneous or non-systematic observation, the teacher makes notes of behaviours that appear important at the time. Systematic observation focuses on one or more previously specified behaviours. These target behaviours are operationally defined and then counted in terms of frequency, magnitude or duration. The latter procedure is used extensively in applied behaviour analysis and for purposes of behaviour modification.

Screening procedures consist of short and easily administered inventories, questionnaires, check lists or rating scales which provide some initial information about the characteristics, i.e. attitudes, feelings, knowledge, or skills, of an individual student or a group in relation to a variety of learning topics or areas. Screening instruments are often constructed by teachers to cover a particular context of learning or student behaviour. Screening can be quite efficient but the accuracy of the information obtained in this way can be unreliable.

Informal interviews have much in common with direct observation. The teacher asks questions and discusses the variables to be assessed with the student, produces notes and might make a report of the interview.

Structured interviews require considerable preparation and planning in advance. The areas focused on may be general or quite specific. The success of this approach depends on the skill of the interviewer in choosing and correctly phrasing the relevant questions, and the ability of the person being interviewed to understand the question and provide answers. An advantage of structured interviews is that they assure that every student is asked the same questions in the same sequence. This facilitates comparison between the responses and reactions of different students and helps to provide a more valid picture of the performance, knowledge, attitudes, etc., of the class as a whole. Structured student interviews are particularly valuable when the teacher's major aim is to evaluate a teaching program or module.

Situational try-outs are based on the assumption that the best estimate of an individual's skills and abilities comes from direct observation of that individual in specific learning or problem solving situations, rather than from a test or indirect report showing what has been learnt in the past. Situational try-outs can provide process as well as outcome information. Peer teaching is the suitable type of situational try-out in the classroom. Students take turns at being the *teacher*. A

considerable amount of information regarding the student's knowledge about how to learn, practise or go about a certain problem is made explicit by this method. Asking more experienced students to explain to their less experienced peers how they would handle a variety of computing tasks forces these *experts* to make explicit what they know and how they go about their work. They could also be asked to teach less efficient *programmers* to produce more effective programs. In devising their instructional plans, students should probably be advised by their teachers. By observing students' questions, learning processes and solution paths, valuable insights into the processes leading to their performance outcomes can be gained. Obviously, after this type of peer teaching adequate debriefing compensating for poor instruction must be provided by the teacher.

Work samples are easily obtained from the students or their parents. The information sought in this method relates to the number, types and patterns of errors as well as successes. Work sample information can be used to develop and modify learning procedures which can correct or circumvent the difficulties experienced by the student.

Analyses of tasks and learning contexts focus on the demands made of the student rather than on the student's skills and personal characteristics. Task analysis identifies the major component skills and appropriate sequencing required to complete a given task. An analysis of the learning context is applicable to any classroom, social, leisure or family setting. It aims to identify the most important characteristics and components in a situation, i.e. the major demands, stresses, obstacles or barriers it might produce for a particular student.

In summary, the following means of monitoring student progress in computing as well as in particular subject areas have been found useful by teachers:

work records, work samples and folders of completed work;
student diaries or journals;
practical performance tasks;
observation and interviews;
observation of peer teaching;
group and peer evaluation;
checklists;
- learning contracts.

PROGRAMMING RELATED EVALUATION

Many methods can be used for the evaluation of a student's or group of students' planning of a computer program. For example, all activities attempted might be listed by the student or students in an activity book. The teacher meets with individuals or groups of students on a regular basis to comment on, accept or ask for revision of the work as it is presented by the students.

Although in computing assignments and computer-based project work the students might be free to choose the order in which they complete activities, it should be made clear, early in the school year or term, that they will need to justify the time that is spent on each activity. Not all students might be expected to complete every activity, but they need to be encouraged to select those activities they choose to complete and then persevere with their selection.

Sources of information which can provide evidence of student performance and achievements include:

- student diaries of thoughts, plans and descriptions;
  descriptions and analyses of techniques used;
  sketches of ideas and products, designs and plans;
  listings of information sources used, with justifications for their selection;
  working models and their modifications;
  errors made and their correction;
  protocols produced from recordings of discussions between students, *thinking aloud* and interviews;
  videotaped records of problem solving activities;

The collection of exemplar, not only of students' programming efforts but also to accompany the student profiles and attainment levels, will assist in comparing the performances of students at different points of learning. Exemplar of students' work can also help teachers who are novices to educational computing to validate their expectations, and they are crucial if we want to avoid a narrowing of the curriculum content and of teaching methods. Exemplar can provide directions leading to more reliable methods assessment and intervention than, for example, statewide testing programs. Exemplar can take various forms, such as

- written case studies, showing a range of a particular student's work and his/her development of learning over time;
  portfolios illustrating the work of a number of individual students on the same topic;
  videos, portraying various learning activities.

TEACHER RECORDS ON STUDENT LEARNING

Having documented information relating to the strengths and weaknesses of individual students, their progress in knowledge and skills in computing and subject areas, and characteristics of cognitive and personal development, the teacher is in a position to develop comprehensive individual profiles for all students in the class. These profiles include quantitative as well as qualitative information, gathered by both formal and informal methods, on the performance and learning processes of individual students.

Table 5.1 Class Record of Informal Comments

	Week Ending								
Student	2.3	9.3	16.3	23.3	30.3	6.4	13.4	20.4	27.4
Frank A.	++	+	+	+++	+				
Mary A.	+	++	+	+	+	+			
Ivan A.		+	++	+	+	+			
Ann D.	+	+	+	+	++	+++			
Michelle E.	++	+	++	+	+	+			

Record sheets for informal comments. Quick, anecdotal comments resulting from observations of individual students can be recorded on file cards, accompanying a classlist on a large sheet of paper, or in a computer file containing a classlist and individual students' records. As all the names are on the same list, it is easy to quickly identify the students who have not been observed. The system might be set up in such a way that there should be at least one comment on each child each week. Comments may be both behavioural and conceptual. An example is shown in Table 5.1. The plus signs (+) indicate the number of comments made on the student in the particular week. Table 5.2 contains examples of records of informal comments on individual students.

Objectives

Whether to fulfil the requirements of a course or to assess whether students have learnt what they were expected to learn, certain objectives must be formulated. Where computing is taught as an integrated course with content areas, the objectives of learning computing should, ideally, be brought together with the objectives of the subject area.

A variety of ways could be used to monitor which objectives have been addressed as teaching proceeds. One way is to state which objectives are being addressed within each unit or topic of work. Another is to list all of the computing objectives, leaving space to write in which units of work have addressed them. Obviously, many objectives will be addressed on many occasions.

Such listings of objectives could be in the form of a table. Alternatively, an objectives wheel, as shown in Figure 5.1, could be used. The completed wheel contains all the learning objectives near the circumference. The learning objectives which have been addressed can be indicated for each unit of work. This type of record makes it easy to see whether any objectives are being

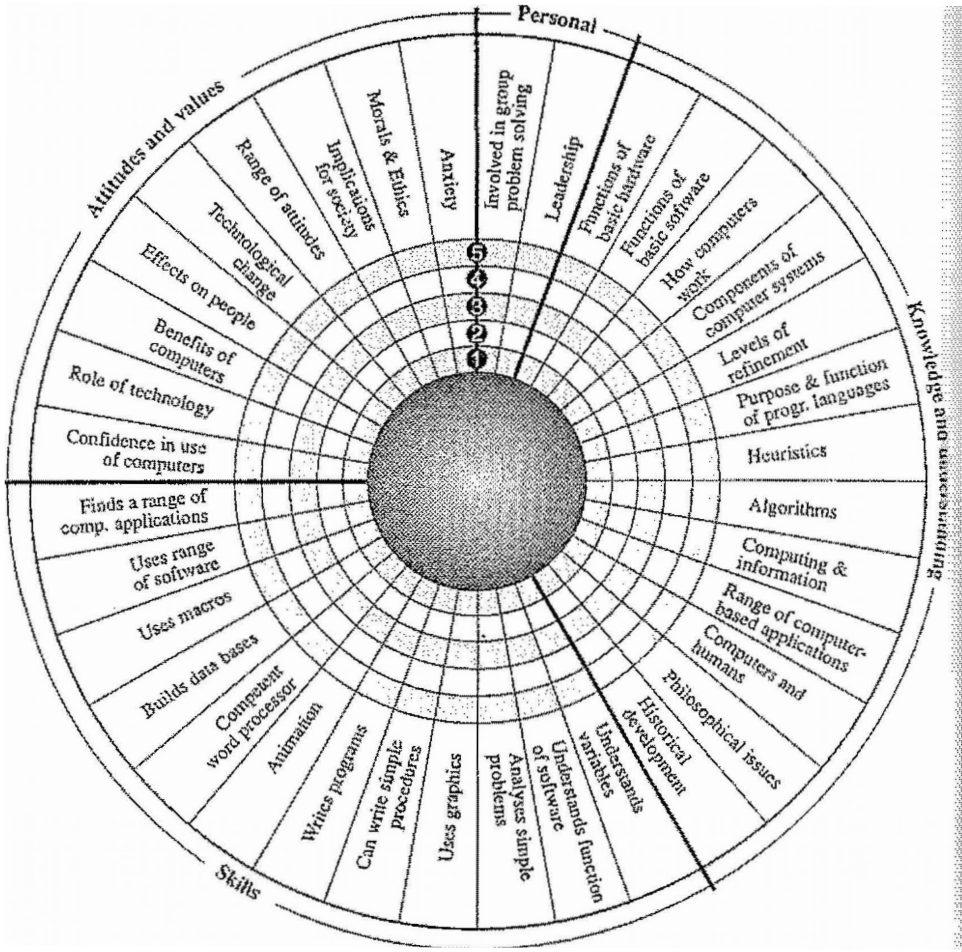
Table 5.2 Sample Records of Informal Comments on Individual Students

Frank A.	Mary A.	Ivan A.	Michelle E.
28.2 Hesitant on his own, but working well with Jim.	1.3 Using 2 hands to type on keyboard.	6.3 Still quiet.	26.2 Participated in decision making in her group today.
2.3 Developing confidence, very keen, tries to do things by himself.	4.3 Very keen, takes work home to complete.	11.3 Can save work.	1.3 Reviews group's work frequently.
6.3 Produced excellent work.	8.3 Prefers to work by herself.	16.3 Keyboard skill poor.	8.3 Creative ideas.
12.3 Able to load program by himself first time.	15.3 Producing excell. work.	22.3 Developing confidence.	12.3 Enjoys problem solving.
20.3 Uses dictionary.	23.3 Helped Ann today.	28.3 Still timid.	16.3 Good organiser, but tends to dominate group.
25.3 Worked continuously for more than 10 min.		4.4 Helped Jane to load program.	
		13.4 Types with both hands now.	

neglected and gives a clear indication of how they are all being addressed. Obviously, the number of wheels required will depend on the number of objectives and the number of units of work.

Objectives wheels could be developed for computing knowledge and skills in combination with different subject domains, such as mathematics, social studies, language, etc. It is possible to place one or more subject areas on one wheel depending on the choice of subjects, or else a number of different wheels could be developed for sets of units within each subject area. A blank wheel is provided for duplication in Figure 5.2.

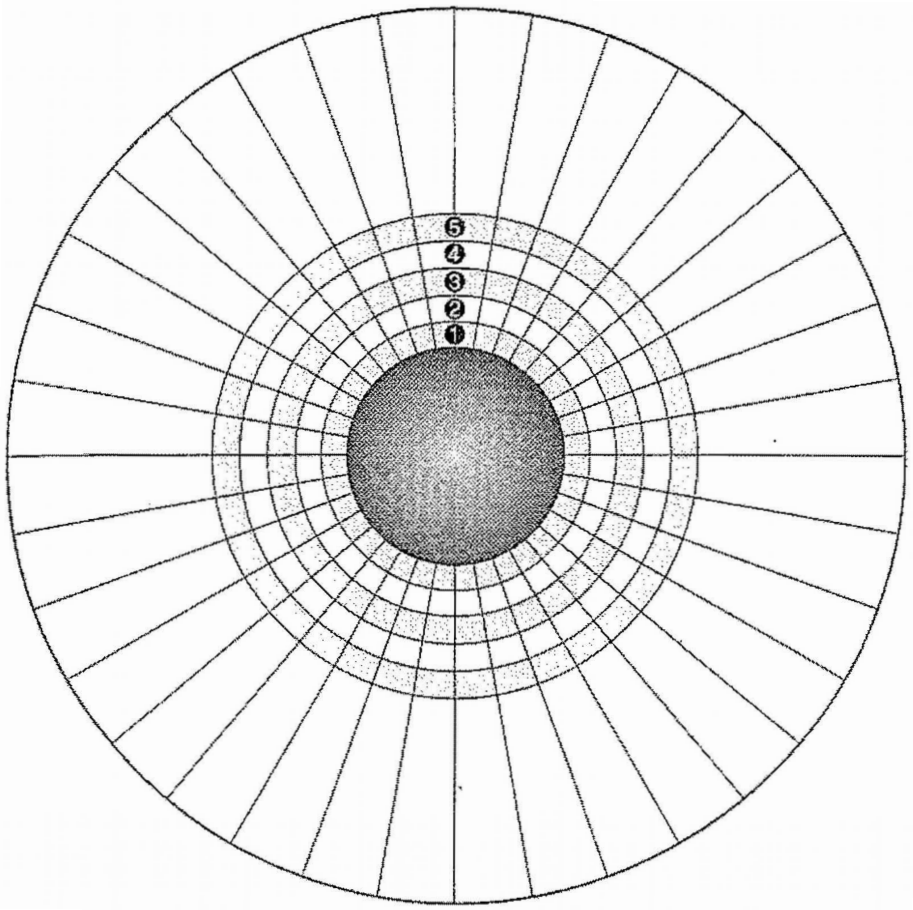
Helping students to become proficient and knowledgeable computer users requires that a balance between practical skills and knowledge about computers and computing be maintained. The actual numbers of objectives covering different areas will be determined by the orientation of the course or syllabus. Obviously, more class time might be spent on some objectives than others. For example, *write procedures* may take longer to achieve than *use a range of computer equipment safely in the classroom* or *identify the components of a typical system.*



UNIT OF WORK

1 .....
2 .....
3 .....
4 .....
5 .....

Figure 5.1 Objectives Wheel



UNIT OF WORK

1 .....
2 .....
3 .....
4 .....
5 .....

Figure 5.2 Empty Objectives Wheel

While all objectives need to be addressed, many will be taught by concentrating on others. It is only necessary to ensure that all are addressed, not that each one has a specific activity related just to it, or that all are given an equal time allocation.

Each time a new unit of work is prepared, it is essential that teachers identify what objectives will be addressed so as to monitor progress towards satisfying the learning objectives.

Tools

Monitoring the use of basic tools available through computing is simple but important. Most teachers' learning objectives would suggest that all should be used. It is easy to neglect one or more tools or to overemphasise one unless there is some formal means of recording the attention given to them. A grid can be drawn up as shown in Table 5.3, and an indication of the unit, part of the unit, or subject area which uses the tool can be noted on such a grid.

Table 5.3 Record of Tool Use

	Tool						
Unit	Data-bases	Spread sheet	Word	Dictionary	Simulation	Program-ming	Impli-cations
History							
Social Studies							
Maths.							
Science							
Language							
Spelling							
Writing							
Reading							
Music							
Projects							

As units of work are developed it is important to record the content that has been covered in such a way that it is easy to identify what remains to be done. This will then assist the development of further units of work. One way is to keep a historical record of the content of a particular unit of work. This is particularly necessary when units cut across multiple topic areas and include parts of the core knowledge and skills of computing. Again a chart would accomplish this.

Listing all the content to be taught and then indicating when it has been presented is another way of monitoring learning objectives. For this purpose a large wall chart, a series of charts or a computer file which is accessible to the students as a *read only file* could be used. An indication of the type of the unit of work could be used beside each tool. This would help to show the links between different subjects. Alternatively, or as well, the date on which the unit was taught could be indicated. Such charts could also be used as devices for planning.

EVALUATION

Evaluation is concerned with educational improvement in the areas of programs, organisation, learning and teaching strategies, assessment procedures, resource usage and all other aspects of the curriculum. A system of evaluation needs to be established and incorporated into learning and teaching computing, whether this takes place under the umbrella of *computer studies* or is integrated into all subject areas of the curriculum. The evaluation procedure needs to be simple and ongoing. The information collected only becomes useful if it is analysed, interpreted and appropriate change decisions based on it are made and implemented. The total process of evaluation is slow and gradual, but it does provide directions for possible change.

Perhaps the most important part of designing assessment and evaluation is specifying exactly what their purpose is. The purpose will determine the scope of the evaluation and direct limited time and other resources towards important targets. The purpose of the evaluation will determine for whom I will collect the information, who will be affected by and informed of the resultant decisions.

The following are some of the questions one might answer *before* committing oneself to certain assessment and evaluation procedures:

1 What am I trying to evaluate?
   - the appropriateness of objectives;
   - the relevance of topics;
   - the effectiveness of teaching strategies;
   - the appropriateness of assessment procedures;
   - student progress.

2 How will I collect the information?
   - questionnaires;

interviews;
observation -- checklists;
diaries -- teaching records, student records;
student projects;
formal tests -- analyse results;
peer assessment;
student's self-assessments.

3 When will I collect the information?
during a lesson;
during the teaching of topics;
at the end of each topic;
at the end of each term or year;
by the end of each week.

4 How will I document the information for myself?
charts;
class records;
records on individual students;
records on groups of students;
description of critical events;
descriptive record of each lesson/day/week.

5 How will I report the collected information?
in a written report;
informally in student interviews;
informally at a staff meeting;
at a parent-teacher meeting.

CONCLUSION

Current standardised tests do not provide information about thinking processes which a student might be using in his/her attempts to solve a problem, so they do not help in the diagnosis of misconceptions or other specific sources of difficulty. For the assessment of computational learning no such tests are available at all. Different assessment procedures are needed which focus on specific higher order thinking skills and which provide information about such thinking processes as the individual learner's representation of the task in hand, his/her construction of a mental model, the generating of hypotheses, the identification of a solution path, the planning of steps towards solution, and the self-monitoring of the application of solution strategies. Such assessment can yield diagnostic information that can help guide instruction that is tailored to the needs of individuals with respect to learning processes and the development of

understanding and knowledge. In addition to alternative assessment that taps cognitive strategies and component processes of learning, there is a need for the assessment of metacognition. This is of particular importance for educationally disadvantaged students. Traditionally, at all levels of the educational system, programs designed to serve students with temporary or more basic learning difficulties have focused on basic skills training and remediation. However, a growing body of research indicates that many of these students perform badly because of an impoverished or underdeveloped repertoire of problem solving, learning and study strategies. To diagnose these difficulties better, we require assessment of the student's employment of metacognitive strategies in planning, monitoring, revising, etc. their learning performance.

Multiple choice format can be used to do more than measure the recall of facts. For example, it can be used to assess the student's understanding of a statement or argument, and to evaluate the student's ability to check the consistency of information, arguments, etc. However, the limitation of multiple choice questions for the assessment of problem solving and thinking processes is obvious. Problem solving skills such as the *identification of reasonable alternatives*, the monitoring of solution processes, etc. require responses constructed by the student. Written or verbal responses are needed not only to assess these skills, but to probe the depth of the student's understanding of arguments, problems, complex situations and his/her ability to understand and think about even more complex messages than are used in multiple choice items. Open-ended test items seem a reasonable way of dealing with the assessment of problem solving, thinking and learning processes, but such testing is very expensive. A compromise could be multiple choice items which include the use of open-ended questions that ask *why* a given multiple choice option was selected and by following multiple choice tests with interviews.

Other methods which have been used successfully in research and could be used in the assessment and mediation of higher order cognitive processes of students learning with computers include interviews by teachers, student diaries, direct observation of student problem solving, student dialogue, role playing, peer teaching. Eventually, computer simulation (focusing attention on the information and strategies which a student uses to solve problems, including the way he/she attacks the task, the number of hints needed, the efficiency of the solution path, etc.) and intelligent tutoring systems can be added to the list of feasible approaches to the assessment of higher order processes. The long-term aim should be a completely integrated learning and assessment environment in which students are given precisely specified cognitive feedback as part of assessment-based instruction.

As we design settings where teaching and assessment are integrated, the instrumental importance of learning as a basis for future learning will be emphasised. Students learn to read so that they will be able to learn through reading and so that they can acquire and interpret information. We learn to write so that we can organise our thoughts and communicate them to others, and so that

we can clarify ideas and build persuasive arguments. In science, students learn to think systematically and critically about our physical and social world, and to ask questions. Students learn computing so that this powerful tool becomes part of their repertoire of instruments, skills and knowledge that allow them to function in and eventually contribute to society as a whole. In focusing on the development of such enabling competencies certain standards will apply, just as they do in current achievement testing, but the types of skills, knowledge and understanding which are assessed, and the way in which knowledge is exercised, will lead to different criteria for evaluation.

PART III

THE EMPIRICAL STUDY

6

Observing Characteristics of Learning with Laptops

Chapters 6, 7 and 8 provide the bulk of information resulting from an empirical study of 115 Year 6 and Year 7 students who have their own laptop computers for use in school and at home. Chapter 6 describes the objectives, design and conceptual framework of this study. An initial attempt is made to investigate *effective* uses of personal computers by students. The chapter also reports on significant student feelings and attitudes to computing, computer awareness and knowledge, all elicited through questionnaires. Finally, it is shown how the students assessed their own attitudes to computers, and computing competence for themselves and others. Chapters 7 and 8 deal with individual differences and gender differences respectively.

OBJECTIVES

The empirical study described here has two major and severe limitations: (1) It is recognised that with increased availability of personal computers, improved software and related curriculum materials and teacher education in computer use in classrooms, the information presented may become outdated quickly; and (2) a study based on observations made on 115 students from two classes in one non-randomly selected school certainly lacks reliability and thus generalisability. Nevertheless, in view of the current large knowledge gap in the area, information on learning and teaching with computers in Australia must be disseminated even if the findings may be timebound. The experiences and reactions of even an unrepresentative sample of students and teachers are not only interesting and informative for teachers and administrators who are planning to embark on similar projects at this stage, but they raise questions that must be addressed in research and in practice.

The objective of the empirical study was to describe and discuss a frequently advocated approach to educational computing as it operated in a *not untypical* classroom during 1991, and to highlight and illustrate some of the issues raised in the chapters attempting to characterise major dimensions of today's state of the art in educational computing in Year 6 and 7 classrooms. More specifically, the

descriptive study focuses on the attitudes, knowledge, abilities and achievements of Year 6 and Year 7 students who work with their own laptops. In addition, the learning practices and preferences of students who are considered by their peers and teachers as particularly *effective* in learning with their laptops, as well as of those students who appear not to be comfortable with using their computers, were observed. In this study the focus was on student characteristics, but some consideration was given to instructional characteristics which might influence individual differences and learning outcomes.

Instruction in the SUNRISE classrooms at Coombabah would be influenced by such teacher characteristics as teachers' attitudes to learning and teaching with computers, their knowledge of computers and computing, their subject-matter knowledge, and their attitudes to the role of computers in society. Though the major concern of the 1991 Research Project related to the students, teacher characteristics are related to student learning processes and outcomes.

DESIGN AND CONCEPTUAL FRAMEWORK

The empirical study sought to describe student learning in terms of certain relationships among several sets of variables:

- student ability/intelligence;
- student feelings, reactions and attitudes (including anxiety and enthusiasm) towards computers and computing;
- student knowledge of computers and computing;
- attitudes to learning and problem solving;
- student perceptions and expectations of their teachers;
- student use of laptops for problem solving and learning;
- developmental differences;
- sex differences.

An initial conception of this system of variables is sketched in Figure 6.1 in which the boxes identify the types of variables and the arrows indicate functional relationships.

Learning context, especially subject domains, teacher characteristics and other unidentified variables, and student perceptions of teacher expectations, are likely to influence student attitudes and learning processes as well as learning outcomes. Attitudes to computing and general intelligence are likely to influence the development of both computer knowledge and subject-matter knowledge. Availability of a family computer at home was expected to influence attitudes and knowledge about computers and computing at least initially.

One might reasonably expect that the greater the student's subject knowledge, the greater the integration of laptop use with learning. In addition, it seems plausible that students who are knowledgeable about computers would use their

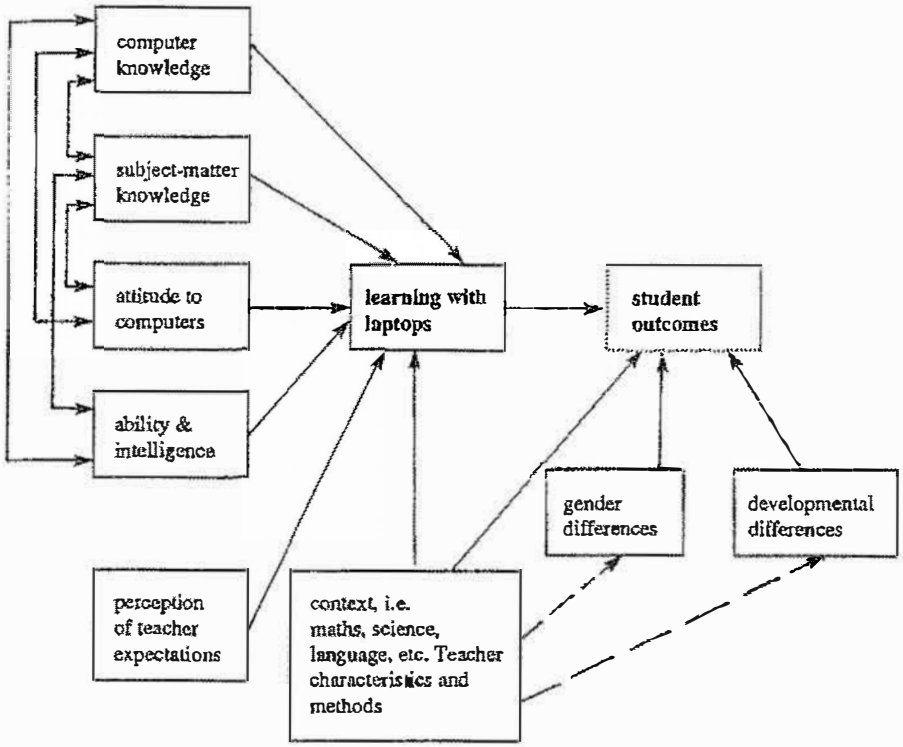


Figure 6.1 Preliminary Conceptual Framework

laptops somewhat differently from less knowledgeable students. The more knowledgeable students, for example, might make fuller use of the computer's capabilities than less knowledgeable students. We asked students how familiar they were with the hardware and software. We also found out whether they served as a resource person or *expert* with respect to certain uses of computers for others in the classroom. The causal links between student characteristics, classroom contexts, teaching and laptop use cannot be disentangled in our data.

Attitudes and knowledge were expected to be interrelated, and ability, perception of teacher expectations, attitude and knowledge were expected to distinguish among patterns of learning with laptops. Ability and educational achievement were regarded as related and were expected individually or jointly to influence computer knowledge, subject-matter knowledge, attitudes to computers and learning with laptops. Finally, different patterns of learning with computers were expected to lead to differences in student outcomes.

DATA COLLECTION

Guiding Principles

The general principles guiding the data collection were that the research be relational, learner-oriented and naturalistic. The major brief from the Queensland Department of Education was that the study describe patterns of learning with laptops and make recommendations for improving resources for teaching, curriculum materials and professional development. This implies that the study should relate patterns of learning practices to characteristics of students and learning contexts. Furthermore, because the study was concerned with learning at school, it was conducted in the students' classroom. This allowed the observation of relationships among variables as they coexist and interact. Only anecdotal information was gathered from parents at a parent meeting.

However, conducting research that is both relational and naturalistic implies some conflict among research goals. Describing classroom phenomena without intruding and changing the nature of the classroom as it is observed requires an unobtrusive, responsive research style. Describing relationships among variables with exactness and precision would have required some control over the research setting.

These conflicts were handled by using a research design that employed both quantitative and qualitative techniques. Data collection procedures emphasised non-reactive measures and systematic observation in addition to formal quantitative measurement procedures. The primary method of data collection was through structured and semi-structured but open-ended interviews. Extensive use was made of questionnaires which contained both open-ended and multiple choice items to elicit information with respect to key variables under investigation. In a number of case studies it was observed how individuals used their laptops in work on given tasks. We also compared students' problem solving efforts with and without computers.

Methods and Sources of Data

Informal interviews with the teachers provided most of the data directly relevant to their views regarding the uses of personal computers in instruction. Teachers were asked about their general instructional decisions and practices, personal use of computers, and their attitudes to lesson planning, presentation and assessment. They were also asked to make suggestions for improving teachers' knowledge and skills in computing and the quality of curriculum materials designed for teaching with computers. Information about the teachers' views on these matters and their experiences and attitudes to computers will be found in Chapters 4, 9 and 10.

Table 6.1 Methods and Sources of Data

Conceptual variable	Method	Source of data
computer knowledge	interview	students
	questionnaire	
	observation	
	structured tasks	
	teacher judgments	teachers
subject-matter knowledge	teacher judgments	teachers,
	tests, interview	students
attitude to computers	questionnaire	students
	interview	
computer use	interview	students
	observation	teachers
	questionnaire	peers
learning decisions and practices	interview	students
	observation	
ability	tests	students
educational achievement	teacher judgments	teachers
	tests	students
	observation	
	interview	
classroom context	observation	students
	interview	teachers
	questionnaire	peers

The interviews, observations, questionnaires and tests provided data on specific indicators of the sets of conceptual variables of interest (see Figure 6.1). The overall data collection effort is summarised in Table 6.1, which identifies the conceptual variables, and specifies procedures used to collect data from participants.

Most interviews were conducted by a single interviewer, Irene Brown. Laurel Bornholt[1] and Helga Rowe conducted some interviews. To establish reliability each of the transcripts of interviews and the problem solving studies were scored by at least two of the researchers. The interviews were structured by Helga Rowe, who also chose and produced tests and questionnaires with the assistance of Isabel Lesman.

Dr Laurel Bornholt from the Education Faculty of the University of Sydney visited Coombabah on two occasions.

To obtain the information necessary to examine this preliminary conceptualisation we collected background information on the students involved in the study, including measures of ability and previous school achievement. This was followed up with a set of structured and open-ended questionnaires, interviews and observational studies of student problem solving and learning. More specifically, for each component in Table 6.1 we collected data on several variables. For example, we administered two different IQ tests, sought teachers' ratings of student achievement in addition to test results, and discussed sample questionnaire responses and observed classroom behaviours with individual teachers.

THE SAMPLE

Total Sample

The study was conducted with 56 Year 6 and 59 Year 7 students from Coombabah State Primary School, at the Gold Coast, Queensland, Australia.

In Queensland students do not start high school until Year 8. The total sample of 115 students consisted of 60 boys and 55 girls. On 30 January 1991 the average age of the sample was 11 years 2 months (Standard Deviation (SD) 8 months). The average IQ estimated by WISC-R was 104 (SD 12). Other characteristics of the sample are summarised in Tables 7.1 to 7.4 in Chapter 7.

Of the students, 62 (i.e. 54%) had had access to family computers at home. 31 boys (i.e. 52%) and 31 girls (i.e. 56%) had access to family computers at home. Whether students had access to family computers at home or not made no difference to the feelings, attitudes and achievement assessed in the present study. Providing each student with his/her own laptop appears to have cancelled out differences resulting from differential access to computers. It is interesting, however, that there was no difference between students whose family owned computers and those who did not at the beginning of the school year in which the students first obtained their laptops.

Year 6 Sample

The Year 6 sample consisted of 30 boys and 26 girls, a total of 56 students. On 30 January 1991 the mean age for the Year 6 sample was 10 years 8 months (SD 4.9 months). The average IQ was estimated to be 106 (SD 12). 32 of the students (57%) had access to family computers at home, i.e. 15 boys (50%) and 17 girls (65%).

Year 7 Sample

The Year 7 sample consisted of 30 boys and 29 girls, a total of 59 students. The mean age for Year 7 students was 11 years 8 months (SD 4.8 months). Their average IQ was estimated to be 103 (SD 12). 30 Year 7 students (51%) had access to family computers at home, i.e. 16 of the boys (53%) and 14 of the girls (48%).

Differences between Year 6 and Year 7

Year 7 students were on average one year older than the Year 6 students and their average IQ was lower by 3 points. This IQ difference accounts for only 20% of one SD of the WISC-R and is thus neither statistically nor psychologically significant. No Year 6 student had previous school computer experience. All of the Year 7 students had been participants of the Queensland SUNRISE Project at Coombabah Primary School during 1990. They thus had one year more experience with computers and computing than the Year 6 students.

EFFECTIVE USES OF LAPTOPS

Selection of *Effective* Users

R1 versus R5 students: Year 6 and Year 7 students were asked to write down the names of the students they regarded as the *best students in computing*, and teachers were asked to rank students in each class according to the effectiveness of their use of laptops. It was suggested that the top most effective students, no more than 20%, be classified in category R1 (i.e. Rating 1) and the bottom 20% be classified in category R5 (i.e. Rating 5). Attempts were made to rank the middle 60% into categories 2 to 4, but the agreement between teachers with respect to these rankings was low. A reason for this was that while R1 students tended to be regarded as *experts* in computing in many if not all areas, and R5 were students who appeared not to cope well with computing, the middle group contained a number of students who were expert in specific areas of computing or had produced one or just a few exemplary pieces of work. Hence the rankings of the middle group were not used in the study.

For the purposes of this study students were classified as R1 if the judgment of teachers and peers agreed. Membership of the R5 group was based on teacher judgments only. The author believes that it is unethical to ask peers to identify the *worst* students. Some of the characteristics of the R1 and R5 groups are provided in Tables 7.3 and 7.4 in Chapter 7.

One aspect of this study was to seek to describe ways in which students nominated as *effective* laptop users used their computers in different subject areas

and more generally. For this purpose we had to enumerate certain attributes of learning with computers. These attributes would then serve as the initial basis for describing patterns of computer use among students. The patterns were later described further (and validated) on a set of additional pedagogical variables. In this section an attempt is made to define these attributes of computer use for learning and to identify measures of them.

ATTRIBUTES OF COMPUTER USE FOR LEARNING AND THEIR MEASUREMENT

An important prerequisite for this part of the study was to decide upon a working definition of the *attributes of computer use for learning*, so that they could be measured and used to distinguish empirically among patterns of learning with computers in different subject domains. This definition focuses on how students integrate computer activities into their schoolwork. This is likely to be influenced by a conscious or unconscious theory of schoolwork and learning in addition to attitudes, feelings, self-image and other less cognitive variables. This conception guides the selection of the attributes of learning with computers, including decisions which students must make about when and how to use the computer, and linking computing with other activities at school and at home, as well as modes of learning. Sensitivity to feedback is of particular importance.

A useful framework for building a definition can be derived from theories of learning and problem solving which emphasise decision making and judgment (see e.g. Shulman & Elstein, 1975; Bruner, 1966, 1987; Bruner & Haste, 1987). This framework can help to define learning with computers because it can suggest specific learning decisions that are likely to be made in relation to computer use and learning tasks in which computers play a role. The basic promise of the decision making approach is that it conceptualises learning as an ongoing process under the active direction of the learner. Learning is multifaceted, containing feelings, attitudes, content, goals, activities and methods orchestrated by the learner in order to provide a flow of activity towards expected and/or not expected outcomes. Students' plans are a central focus of this conceptualisation. In formulating and evaluating plans, learners integrate information about the learning task, the context within which it is presented or presents itself, the teacher's behaviours and expectations, and peer and other adult models in order to reach judgments or decisions that guide learning processes. Also, learners can monitor ongoing processes. If problem solving and learning activities proceed as planned and feedback appears to be positive, the learner concentrates on maintaining the current direction. If activities are not going according to plan or some disruption occurs, they activate a routine for handling the disruption. A final aspect of monitoring is when students evaluate the outcomes of their work in order to come up with an improved plan.

Defining Attributes

In this attempt to define attributes of computer use for learning, the assumption is made that computer use fits within students' ongoing planning and decision making. Another assumption is that students can make reasonable choices among alternative approaches to learning for reaching one or a combination of learning goals, and among the modes of using computers (e.g. drill and practice, construction, simulation), given their knowledge of the subject area in which the task is presented, their knowledge of computing, their own skills and experience in the area, and perceived teacher expectations.

One might assume that *effective* learners with laptops will make reasonable decisions about using their computers and the software available to them to accomplish their learning goals, the structure of their learning processes, the use of their subject-matter knowledge in the domains of their curriculum, and that they make use of the advice of teachers and peers and other resources in the classroom. Once planning decisions are made, the learner tries to carry out the plan. Finally, *effective* learners must be able to monitor what they are doing, be sensitive to feedback, take remedial steps when warranted, and evaluate their own performance.

This conceptual framework implies that success in learning with computers depends strongly on students' planning and decision making that is linked to self-monitoring, evaluation and feedback-based repair strategies. In general, this framework suggests that the use of computers by students in their daily learning activities should be defined as the degree to which computer activities are integrated into planning processes for learning, in the sense that computing is related to other learning strategies and tasks. A reasonably general definition of effective computer use by learners which takes into account the elements of planning, monitoring and feedback might be as follows:

> *Effective computer use by learners involves the appropriate integration of computing activities with the ongoing curriculum and personal learning goals, which can change and become refined on the basis of feedback that indicates whether desired outcomes or part-outcomes have been achieved.*

The above definition contains a number of conceptual components, i.e. learning goals, ongoing curriculum, computer-based activities, appropriate integration, and feedback. These are explained in Table 6.2. Each component in turn contains specific, measurable indicators, which, together, might constitute an operational definition of certain attributes of learning with computers. These attributes and their indicators are briefly described in Table 6.3.

Learning goals. The focus on the personal goals of learners is a direct result of choosing a decision making framework which assumes that students' behaviours are purposive, i.e. goal oriented. In order to determine the appropriateness of learning strategies used to meet these goals, teachers must endeavour to

Table 6.2 Components of Effective Computer Use by Students

LEARNING GOALS

- a Achievement
  - 1 Mastery of basic skills and procedures
  - 2 Acquisition of subject matter concepts
- b Motivation
- c Social

ONGOING CURRICULUM

- a Subject matter
  - 1 Content analysis
  - 2 Major topics
- b Course materials
  - 1 Tangible/Demonstrations
  - 2 Information sources (books, disks, manuals)

COMPUTER-BASED ACTIVITIES

- a Modes of computer use
  - 1 Drill and practice
  - 2 Computer tutorial
  - 3 Simulation
  - 4 Microworlds
  - 5 Games
- b Learning in pairs or groups
- c Imposing time restrictions on oneself

APPROPRIATENESS OF INTEGRATION

- a Contribution of computer to learning goals
- b Coordination between the curriculum and computing
- c Strategies for selecting computer activities

FEEDBACK

- a Monitoring of own progress
- b Comparison with class standards (performance of peers or teacher expectations)
- c Changes in computer-based activities
- d Direct feedback from teacher or *expert* peer

understand the nature of the objectives which different students wish to accomplish.

Learners' goals may include outcomes that are academic, interest-based and hence motivational, social, or some combination of these. Academic goals include the acquisition of computing or subject-matter-based knowledge or skills. Motivational goals include increase in personal interest in the subject-matter and positive attitudes to learning, teachers, and the learning environment generally.

Indicators of *Effective* Laptop Use

	Source of Information	Definition
LEARNING GOALS		
mastery	interview	degree to which student uses laptop to reach basic skills goals
cognitive	interview	degree to which student uses the computer to help in problem solving and learning
motivation	interview	degree to which student uses the laptop for self motivation
management	interview	degree to which student uses laptop to maintain a system for his/her work
computing	interview	whether student views use of laptop as a goal in itself
ONGOING CURRICULUM		
learning facts	interview	how extensively does the student use laptop?
coordination	interview	how well does student coordinate computing with other learning activities?
integration	interview	student rating of extent to which he/she has integrated laptop work with other activities
non-school learning	questionnaire	whether student uses computer for more than school work
COMPUTER-BASED ACTIVITIES		
N of modes	questionnaire	number of different modes of learning used on laptop (e.g. drill and practice, writing, database, simulation, computation, problem solving)
social preference	questionnaire	does student prefer to work alone, in pairs, groups?
time restriction	questionnaire	does student impose time restriction for laptop use on himself/herself?
APPROPRIATENESS OF INTEGRATION		
perc. success	interview	self-rating of success in using laptop for learning
success	teacher judgm	teacher rating of success
FEEDBACK		
change use	interview	does student modify strategies on the basis of computer-based feedback?

Social goals relate to cooperation or team work among students. One of the most complex tasks faced by the teachers is that of balancing the multiple goals within a lesson. Personal computers add an extra order of complexity into this balancing act.

To understand learning with computers, we need to determine the absolute and relative importance of these learning goals for the students. Questionnaire items and interviews with individuals identified some of the learning goals for computing and other subject-matter, and the degree to which the computer is perceived to facilitate the achievement of the latter. Specifically, the questionnaires and interviews sought information about the extent to which students felt the laptops helped them master basic skills, acquire new concepts and knowledge, increase their liking of curriculum subjects, and increase their motivation for learning in different domains. This information provided the basis for a rating of the extent to which students perceived the laptop to have helped them reach their goals. The measures are defined in Table 6.3.

Ongoing curriculum. The student's goals are pursued in the context of an ongoing curriculum that is communicated through a variety of teacher-initiated activities and demands. Shavelson & Stern (1981) define the curriculum to include: (1) subject-matter, i.e. the major content areas and important concepts that are taught within each content domain, and (2) materials, tools, etc., i.e. the things that students observe and/or manipulate (drawing equipment, calculators, set exercises) as well as textbooks, maps, etc. These components are important to note because they define the range of activities in which the laptops can potentially be integrated. For this assessment, use of laptops was viewed in relation to students' planning decisions for coordinating laptop use with the various activities occurring in the classroom.

Computer-based learning activities. This component relates directly to the learning technology. The decision making perspective suggests that during planning, students will make important decisions, i.e. choices among possible computing activities.

One important distinction can be termed the *mode of computer use.* This refers to the selections computer users make among forms of available applications, such as drill and practice, note taking, calculation, databases, games, microworlds, etc. A second dimension relates to whether students prefer to work by themselves, in pairs or in small groups. A final distinction relates to the allocation of time. Students might allocate themselves time for certain activities or for total computer use in, say, a subject area, a morning or an evening.

Interviews and observations provided information on the different modes used by students, and the number of different modes used by individuals served as a measure of variety of computer uses. The integration of the computer into the students' learning tools was measured in terms of the *flexibility* with which students used their laptops. For example, students with numerous learning goals in a range of subject areas would put the computer to a variety of alternative uses. In addition, information was collected on the student preferences for working alone, in pairs or in groups of more than two students.

Appropriateness of integration. The various components described above come together in considering the integration of computer use with learning, and the appropriateness of the various forms of integration. Integration of computing into learning activities can occur with respect to personal learning goals as well as the curriculum. For example, the fact that students have numerous personal learning goals implies that the computer might be put to a variety of alternative uses. In a sense, appropriateness of integration implies appropriate, flexible and well orchestrated use of the computer as an important instrument, tool or set of tools.

Feedback. The decision model of learning indicates that students' evaluation of their own work and, if necessary, modification of learning strategies relative to their goals is an important part of learning.

PATTERNS OF LAPTOP USE

The operational definition of personal computer use for learning identifies a number of dimensions of laptop use, but it is not yet clear how independent these dimensions may be of each other. If they are closely related to one another in practice, it would be possible to identify a continuum of practices and to order students along that continuum. In other words, if students who establish multiple goals for laptop use are also those who integrate and coordinate the laptop with the curriculum and other classroom activities, use multiple modes of learning appropriately, and modify their learning practices based on feedback, distinctions among conceptual dimensions would not be as important. A single dimension might then be used to describe students' use made of laptops in learning.

Alternatively, some students might integrate and coordinate laptop use to master basic skills, using it mainly for drill and practice activities, while others might use the computer for enrichment, using a variety of learning modes with only loose coordination of laptop use with other classroom activities. Yet others might use the computer solely for word processing and/or as a calculator. In this case the conceptual distinctions among the dimensions would be important, and the manner in which the students use their computers could not be described simply on a single dimension. Rather, patterns of use, some possibly more valued than others by teachers, would describe the differences among learners.

In order to examine the underlying relationships among the measures of the attributes of laptop use defined in Table 6.3, some correlations were calculated between them. The results tended to point to a multidimensional rather than a unidimensional interpretation of successful laptop use. All but two of the correlations were less than .20 in absolute value. Coordination and learning facts correlated.

Grouping of Students by Laptop Use

Recognising that *effective* students used their laptops in more than one way, an attempt was made to devise an initial set of categories with which to describe

patterns of laptop uses by learners. The author recognises the potential pitfalls of creating categories and labels, especially on the basis of largely qualitative data and for the description of a rapidly evolving phenomenon such as learning with personal computers; but her belief is that the benefit of possibly capturing variations in learning patterns among *effective* laptop users on the basis of some initial groupings outweighs the potential harm of creating an image of learning with computers that would stifle its evolution. The grouping serves as a short-term initial snapshot of descriptive information which may or may not prove useful to teachers and software developers. Everyone needs to keep in mind that educational computing is changing rapidly and that students learning in environments such as the Coombabah one are changing even more rapidly.

Methodologically, we used profile matching, as the sample was too small for cluster analysis (see Shavelson, 1979), to group together students with similar repertoires of use, and attempted to distinguish them as clearly as possible from students with other repertoires. On the basis of this qualitative matching four possible groups were identified. Group A (n=22), for the purposes of this study labelled *orchestrators*, represented the widest variety of learning applications closely linked to teacher and task demands, personal aims and skills, personal learning style and the current social demands of the classroom. Their attitudes to and uses of their computers reflect a harmony created by appropriate and flexible application. They made the computer part of themselves. Group B (n=31), labelled *amplifiers*, capitalised on available software and procedures written by others, and as a consequence, integrated computing with subject-matter and other classroom activities least of any of the groups. They use the computer to amplify their existing skills, but view it as an adjunct, i.e. a non-essential but at times useful or convenient accessory. Group C (n=26), here labelled *machinists*, used the computer to selectively augment learning in mathematics (used as calculator) or writing (used as typewriter). When pressed they would use whatever Logo options were in their repertoire with varying success. Group D (n=23), labelled *perseverators* tended to make use of procedures and programs written by others. They used these over and over again, and spent much time on the same task or activity. Group D used the computer mainly for drill and practice, particularly in practising spelling and multiplication tables. The groups are described further in the next section.

Orchestrators. This label was applied to Group A because students in this group appeared to be able to intertwine learning goals, subject knowledge and basic skills with computer knowledge, skills and use in harmony with each other. They used all their knowledge and skills flexibly and appropriately to meet a variety of task demands in different curriculum and leisure areas. These students can coordinate computing with textbooks and other instructional aids. They are sensitive to feedback and reflective. They have a good self-image of themselves as students, and show a positive attitude to computing, learning in curriculum domains, their peers and teachers. Group A students report that computing is useful, interesting and gives them control over their own learning. Not surprisingly, the majority of students rated by peers and teachers as the most *effective* laptop users in their classes showed Group 1 profiles.

Amplifiers. Group B students regarded the computer and computing as another area of learning, i.e. an instrument which amplified the curriculum. These students were certainly prepared to use the computing procedures and software components supplied to them for much of the ongoing curriculum and/or to master basic skills. They used their computers less than the students in Group A and they tended to regard and use computing more as an end in itself than as a tool which helps accomplish goals. On the whole, students in this group were less positive in their attitude to computing. In particular, they were more critical than the students in groups A, C or D of teachers' lack of detailed knowledge of some aspects of the technology. These students believed that teacher should have all the answers.

Machinists. The students in Group C tend to view the computer as a non-essential accessory. They felt that the computer is most useful in mathematics and for writing and thus used their laptops largely as calculators and typewriters. These students used their computers more selectively than students in Groups A and B and less frequently. They seemed not to have changed the way in which they use computers from the time they started using them.

Perseverators. The number of students in this group, Group D, was small. These students used the computer for a very limited range of purposes and they tended to repeat the same task or work at the same content for long periods of time. They often copied procedures developed or used by other students in the class and used them over and over again. Group D students appear not to be interested in developing higher order skills for themselves, they are perseverating users of the ideas of others. They were very happy when instructed to use the computer for what is usually termed *drill and practice*. Students in this group appeared not to develop personal learning goals and plans.

Indications of Patterns

Patterns of laptop use varied among the sample of 115 students. Roughly 19% were *orchestrators*, 27% used their laptops for amplification and enrichment, 23% were *machinists*, and 20% appeared to use their laptops for drill and practice only. Thirteen students, i.e. 11% could not be classified within the limitations of this framework.

The variations between students in their use of computers, which are here reflected by group membership, may be associated with student characteristics such as their attitudes to computers, their knowledge of the subject-matter, their knowledge of computers and computing, personal goals and experience with computing, their general intellectual ability and educational achievement. Variations may also be associated with student-teacher relationships, feedback from teachers, student perceptions of teacher expectations, reinforcement

provided by teachers to individual students and the interaction of any or all of these with temporary or more permanent environmental factors. Variations may also be associated with low reliability of student responses to questionnaire items, bad sampling of observations and other variables related to the approach of the investigators.

Individual differences among students in their attitudes and knowledge, however, do not and cannot account for all the systematic variability in learning with computers. There are instructional, curriculum and contextual factors that encourage, discourage, or set limits on the kinds and range of purposes for which individual students employ laptops. Perceived teacher expectations and policies also influence computer use by students. For instance, class *experts* were always *orchestrators*. The students who complained that only selected peers were given instruction in certain procedures so that they could become class *experts* and teach others tended to be in Group D.

Classroom Organisation

The way in which classroom teachers implement an innovative educational program has profound effects on that program's impact and longevity. According to the literature (e.g. Berman & McLaughlin, 1978), the essential ingredient of a successful implementation is teacher support. Means of achieving the latter are discussed in Chapter 9. The organisation and composition of students in the classroom profoundly affects instructional processes and outcomes (e.g. Borko, Shavelson & Stern, 1981; Burstein, 1980; Barr & Dreeben, 1977; Walberg, 1976; Webb, 1980). To what extent are variations of student learning with computers related to classroom organisation and student composition? Was Lipkin (1982, p. 7) correct in warning that 'the urban, low-income minority student . . . is more likely to be provided with drill and practice . . . while middle class students are more likely to use it for more creative purposes relating to problem solving and discovery'? These variables could not be investigated within the constraints of this empirical study. There appeared to be little variation in SES and there were very few children from minority backgrounds in the classes at Coombabah.

Ability level was associated with variation in laptop use. Specifically, students of above average ability tended to be *orchestrators* and seeking the acquisition of new skills and concepts which they could integrate with their knowledge of Logo. As the ability level decreases, laptop use tends to be for amplification and as *machinists*. Students with low ability tended to be *perseverators* (i.e. members of Group D) and use the computer mostly for drill and practice.

Although there is substantial evidence that low achieving students need instruction and practice in basic skills, if this is all they receive in relation to computing, their encounters with computers clearly distinguish them from average or above average students. Put another way, students of low ability (and minority group students) might well get the message that computers are there to drill them while other students might learn that the machine can serve a variety of functions including programming or as a tool for problem solving, depending on the student's goals and needs. The important challenge for low ability students, as for everyone else, is not learning how to use the latest piece of hardware or software but asking how and when it should be used.

OTHER STUDENT CHARACTERISTICS

Feelings and Attitudes

The feelings, attitudes, knowledge of computers and computing, learning and problem solving, perceptions and expectations of teachers, etc. reported in this section were measured on the basis of a series of questionnaires. The questionnaires were administered twice: in April 1991, i.e. two months after the children in Year 6 had obtained their laptop computers and approximately 14 months after the students now in Year 7 had obtained theirs, and again at the end of November 1991, when the school year came to a close. A few items were altered in the second administration. Consistency of responses was tested for both the April and November data by correlating the student responses to parallel items which were either phrased slightly differently or asked in reverse. The consistency of responses for these items ranged from .60 to .87 in November and from .68 to .82 in April.

Students nominated as unusually successful learners with computers held uniformly positive attitudes regardless of their other interests, sex or year. Indeed, 60% of all students were extremely positive towards computers and indicated that all schools should provide computers which students can use at all times. Over 90% of the sample stated that personal computers are as important to students as textbooks. In April more than 80% of the students (in November 70% of Year 6 and 65% of Year 7) indicated that once they start working on their computers they find it hard to stop.

Underachieving and low ability students. A popular myth is that underachieving and low ability students have negative attitudes towards computing. This simply was not true in the sample of 115 students in the current empirical study. 77% of the students of below average ability and achievement reported that *using computers is fun* and 23% found it to be fun at least *sometimes*. 85% of these students were very interested in learning about computers and regarded computers as as important as textbooks. The enthusiasm for computers and computing of the students of low ability and achievement did not differ from that of their peers either in April or in November. Computer anxiety was as prevalent amongst them as in the higher achieving students, but the level of computer knowledge and skills of students of below average intelligence was lower than that of their peers with higher IQs.

Underachievers. This term is often used to describe students of average or above average IQ whose school performance and general educational achievement are below average. On the basis of this criterion, five per cent of the students in both years could have been classified as underachievers. We analysed questionnaire responses, data obtained in interviews and observations of these students and found that they showed very positive attitudes to computing. They showed higher levels of effort, intrinsic motivation and task interest when working with computers than in their work without computers. It certainly appears that computing in the classroom might be used to maximise the underachievers' motivation to learn. Two of five underachievers in Year 6 were judged by peers and teachers to be performing very well in programming, and were selected into the R1 group. Computing can provide extra motivation for these and other students and thus enhance their general academic performance. However, the effectiveness of efforts to coordinate the computer and schoolwork to maximise learning outcomes for underachieving students is unclear. The underachievers were not found among the *orchestrators*, and rarely among the *amplifiers*. An explanation for this finding may lie in the restricted teaching styles applied in these classrooms, or it may be an effect of specific requirements of Logo programming. Both have implications for integrating computing with instructional and learning goals.

Anxiety In April 40% (37% of males and 44% of females) in Year 6 and 54% (63% of males and 45% of females) in Year 7 reported that the computer does not scare them at all. In November 30% more of the students in Year 6 and 20% more of those in Year 7 were not scared at all. The increase in students not scared among Year 7 girls was 26%. This means that through being accustomed to the computers some fears are alleviated. However, in both classes there are 25 to 30% of students who are still scared of their computers.

In April 51% of Year 6 students (57% of boys and 44% of girls) and 39% of Year 7 students (37% of boys and 41% of girls) felt that *using personal computers makes students nervous*. The prevalence of this feeling was reduced by 30% for Year 6 and by 20% for Year 7 in November. In April 56% (57% of males and 56% of females) of Year 6 and 25% (30% of males and 21% of females) of Year 7 students reported that their computers used to scare them. In November 62% (60% of males and 64% of females) of Year 6 and 37% (31% of males and 43% of females) of Year 7 reported that they had initially been scared of their computers. The increase in the perceived anxiety of the Year 7 females is likely to be the result of an increasing decline in feeling comfortable with computing in this group. They report not to be coping, they feel that they have been left behind and that they are now so far behind, especially in procedures, that they can no longer catch up.

93% of Year 6 and 78% of Year 7 students reported in April that when they first had them they were frightened that they may break their computers. This fear was reduced by 20% for all groups by November. 65% of Year 6 and 36% of

Year 7 students reported in April that they are still frightened that they might break their computer. In November this fear was reduced by 30% for Year 6 and by 10% for Year 7. One quarter of the students are still frightened that they might break their computers. This group may be made up of the generally more anxious children. On the other hand, these students may show a healthy concern for expensive property or they may fear that a damaged computer might not be replaced for them.

Enthusiasm All students but three Year 7 girls reported in April that they liked using computers. All males expressed the same enthusiasm in November but only 96% of Year 6 and 93% of Year 7 girls expressed this. In April 84% (83% boys and 84% girls) of Year 6 and 80% (87% boys and 72% girls) felt computers are fun. In November 85% of Year 6 and 70% of Year 7 (86% boys and only 54% girls) students felt computers are fun. Year 7 students are becoming more doubtful about the joys of computing, especially Year 7 girls.

In April 87% of Year 6 and 83% (87% of boys and 79% of females) of Year 7 students reported that once they start working on the computer, they find it hard to stop. By November these percentages were reduced for Year 6 to 71% and Year 7 65%. Even Year 6 students who have been using computers for a year less than Year 7 and have not been part of the traumatic beginnings of the project during 1989 (cf. Ryan, 1991) are less keen at the end of the year. Another explanation of this reduction of the initial enthusiasm may be that the students may be approaching a more realistic view of computing. However, if this were the case why would Year 7 students also become less enthusiastic? One would have expected them to have become reasonably realistic about computing by the end of their first year of laptop experience. Computer users in the outside world do not appear to be becoming less and less enthusiastic over the years of computer use.

In April only one girl in Year 6 and 20% of boys and 24% of girls in Year 7 wished that they did not have to take their computer home at night. In November 13% of boys and 20% of girls in Year 6 and 24% of boys and 43% of girls in Year 7 wished that they did not have to take their computer home every night. This is another indication of the decrease in enthusiasm and lessening of interest particularly of Year 7 girls. This contrasts with the finding that there was no significant reduction between April and November in feeling glad to be learning to use computers (96%-100% in all groups), and in the judgment that most students in the class really enjoy using the computers (90% and more).

Nearly all Year 6 and Year 7 students disagreed with the statement *I would be happier to be in a class where they do not use computers* in April. The view of Year 6 students did not change during the school year, but 17% of males and 11% of females in Year 7 stated in November that they would be happier in classes where no computers are used. In April 85% of Year 6 and 81% of Year 7 stated that they would be very unhappy in a school where there are no computers. For Year 6 the feeling did not differ in November, but for Year 7 the enthusiasm

is reduced significantly. Only 67% (72% of boys and 61% of girls) would feel unhappy in a school without computers. The perception that computers make more mistakes than clever humans is slightly down from April in both classes.

Support for the statement *When I am an adult I will get a job where I can use computers* was slightly higher in November for Year 6 (76% vs 70% in April) and lower for Year 7 (58% vs 64% in April), again mainly as a result of the decreased interest of Year 7 girls.

In April Year 7 students were probably more knowledgeable and thus realistic than Year 6 students. Then 83% of boys and 100% of girls in Year 6 and 73% of boys and 72% of girls in Year 7 felt that the computer is their *friend and helper*. In November 87% of boys and 88% of girls in Year 6 and 83% of boys and 75% of girls in Year 7 felt this way.

Self-image and Confidence In April 62% (73% of boys and 48% of girls) of Year 6 and 71% (77% of boys and 65% of girls) in Year 7 agreed with the statement *When I am working with computers I know I will get the work done well*. In November agreement increased to 78% (80% of boys and 76% of girls) in Year 6 and 79% (86% of boys and 71% of girls) in Year 7.

In April only 10% of Year 6 and 20% of Year 7 students believed themselves to be *not the type to do well with computers*. In November this lack of confidence increased to 18% and 32% for Years 6 and 7 respectively. At the beginning of the school year nearly all students agreed with the statement *I am sure I will do better and better with computers*, but there was a significant reduction in this confidence over the year in Year 7 boys and girls.

In April most students in both years felt that some students in their class are finding computing more difficult than they do. There was a significant decrease in this view in Year 7 in November. The confidence of these students in their own computing is waning. And yet, in November 53% of Year 6 (47% of boys and 60% of girls) and 79% of Year 7 (86% of boys and 71% of girls) still believed that personal computers will make better thinkers of students; in April 64% of Year 6 and 81% of Year students held this view.

By the end of the 1991 school year 62% of Year 6 and 70% of Year 7 (76% of males and 64% of females) believed that *Every schoolchild and adult should be able to use personal computers*. In April only 36% of Year 6 and 52% of Year 7 had agreed.

Knowledge of Computers and Computing There was a significant increase (13%) in Year 7 males feeling that computers are not as intelligent as people. Year 6 felt this less strongly than Year 7.

In April 51% of Year 6 and 95% of Year 7 regarded the computer as a tool just like a hammer or a lathe. By November Year 6 agreement increased by 20% for both boys and girls and Year 7 remained much the same.

About 33% of all the students did not believe, in November, that one can use computers without understanding how they work; in April 45% of Year 6 and

54% of Year 7 held this view. Year 7 boys in particular are realising that the further they get in their computing the less frequently they will be able to use the computer routinely, i.e. without understanding. After only one year of laptop use Year 6 have not yet come to this realisation.

In April 89% (93% of boys and 84% of girls) in Year 6 and 73% (80% of boys and 65% of girls) in Year 7 felt that *Using the computer makes mathematics learning easier*. In November 10% more Year 6 students but 10% fewer Year 7 students (16% fewer Year 7 girls) agreed with this.

Children who use computers do better in their schoolwork. In April about 20% of Year 6 and only 10% of Year 7 felt this would never be true. In November there was no change for Year 6 but Year 7 disenchantment with the tool increased to about 20%. Year 7 females did not change their mind: they still believe in November as strongly as in April that students who use computers will do better at school. As will become evident, Year 7 girls in this sample are less realistic about what computers might do for them than their peers and the Year 6 students. They are also clinging to the view that their laptop might help them from failing in their school work.

Most students students believe that *Personal computers are as important to students as textbooks*. There was a strengthening of this belief between April and November for Year 6 and a weakening for Year 7. Also, agreement with the statement *Learning to work with computers is just as important as reading, mathematics and spelling* in April was 81% for Year 6 and 83% for Year 7; in November this support decreased to 76% for Year 6 and 67% in Year 7 (only 61% of Year 7 girls agreed).

Learning and Problem Solving In April more than 70% of students in both years believed that using computers makes learning more difficult. This proportion was reduced by 30% for males and Year 6 females and by 10% for Year 7 females. In April 89% (93% of boys and 84% of girls) in Year 6 and 73% (80% of boys and 65% of girls) in Year 7 felt that using the computer makes mathematics learning easier. In November 10% more Year 6 students but 10% fewer Year 7 students (16% fewer Year 7 girls) agreed with this. This finding was supported by the results of a similar item referring to *schoolwork* rather than to *learning*. In April 75% of Year 6 and 74% of Year 7 felt that at least sometimes using computers makes schoolwork more difficult. In Novembers these percentages were reduced to 46% for Year 6 and 65% for Year 7, though 79% of Year 7 girls in November believe that to be at least sometimes true.

There was no change in the students' response to the statement *Computers are better than books* between April and November for Year 6 (85% yes) but a 13% decrease for Year 7 (90% to 77%), also a 20% decrease for Year 7 girls.

In April 89% (93% of boys and 84% of girls) in Year 6 and 73% (80% of boys and 65% of girls) in Year 7 felt that using the computer makes mathematics learning easier. In November 10% more Year 6 students but 10% fewer Year 7 students (16% fewer Year 7 girls) agreed with this.

In April about 20% of Year 6 and only 10% of Year 7 felt the computer would never be the cause of a student doing better in schoolwork. In November there was no change for Year 6 but 20% more Year 7 students than in April held this view. Year 7 females, however, did not change their minds: they still believe in November as strongly as in April that students who use computers will do better at school.

In all groups 90% and more of the students would like to know how the computer works. No difference in this curiosity was observed between April and November.

Even though Year 7 students are becoming somewhat less excited about computing, a larger number of them than Year 6 understand (even in November) how some people can spend so much time with computers and enjoy it. 90% of Year 7 males (a 30% increase from April) can understand this. One might infer that the enthusiasm for and appreciation of computers and computing is still very strong in these students but that they feel limited in what they themselves can do. More focused teaching might be an answer.

In April 76% of the students in both years were reading books about computers and computing. November showed decreases of 16% and 23% for Year 6 and 7 respectively. In April 36% of Year 6 and 39% (57% boys and only 21% girls) tried to find books about computers and computing in the library. In November there was a 14% increase in this for both boys and girls in Year 6 but a decrease of 12-14% for Year 7. In April 76% of the students in both years were reading books about computers and computing. November showed decreases of 16% for Year 6 and 23% for Year 7.

Figuring out what went wrong in my computing is interesting. In April 71% of Year 6 and 47% of Year 7 agreed with the statement. In November 36% of Year 6 and 35% of Year 7 agreed. This time it was not due to Year 7 girls only, though their agreement dropped by as much as that of Year 6 boys and girls (27-30%). This finding suggests that students may not have been given sufficient help and/or feedback from teachers when figuring out where they went wrong.

Children who use computers do better in their schoolwork. In April about 20% of Year 6 and only 10% of Year 7 felt this would never be true. In November there was no change for Year 6 but Year 7 *never* responses increased to about 20%. Year 7 females did not change their mind: they still believe in November as strongly as in April that students who use computers will do better at school.

In April 51% of Year 6 and 63% of Year 7 indicated that having got used to working with computers they would find it difficult to work without them. In November the attitude of Year 6 had not changed substantially, but 20% fewer Year 7 students felt they would find it difficult to work without computers. Observations showed that the Year 7 students and some Year 6 are becoming frustrated with their lack of ability to control their computers. A considerable number of students are starting to do things without their computers. The novelty has certainly run out.

A popular myth suggests that providing students with laptop computers will reduce, if not stop social interaction and collaborative learning in the classroom. The 115 students in the current sample were unanimous in their view that using laptop computers does not stop students from discussing work with other students.

Control An important aspect of laptop ownership and other modes of easy access to computers for students relates to the *control* students perceive themselves to have over their own learning processes. Approximately 80% of the students believe that using laptops puts them in charge of their own work. There was a 20% increase of this view for Year 7 between April and November, no difference for Year 6.

In April 75% of Year 6 and 80% of Year 7 students felt they had control of what they do when they use the computer. There was an increase in this view in excess of 10% (20% for Year 6 females and 14% for Year 7 females) by November. Learning has taken place. Students are developing their skills and are becoming surer of themselves in most areas. In April 98% of Year 6 and 90% of Year 7 saw themselves as *the boss of my computer*. In November there was a 10% decrease in this attitude, showing either that the students are becoming more realistic about their expectations of success with computing, or that some of the weaker students have experienced failures and disaster. This point is discussed further in the analysis of the feelings of students in different IQ categories.

62% (73% of boys and 48% of girls) of Year 6 and 71% (77% of boys and 65% of girls) in Year 7 reported in April that when they are working with their computers they know that they will get their work done well. In November the proportion of those holding this view increased to 78% (80% of boys and 76% of girls) in Year 6 and 79% (86% of boys and 71% of girls) in Year 7.

Favourite Activities Using Computers In April 27% of Year 6 students chose subject specific and 73% programming activities, in Year 7 32% chose subject specific and 68% programming activities. In November, however, in Year 6 the subject specific preferences increased by 12%, Logo programming preference decreased by 26% and using the computer to play games increased by 14% (from zero in April). For Year 7 in November both subject specific and programming preferences decreased by 6% and 5% respectively, while play increased by 11% (from zero in April). An interesting observation is that in Year 6 significantly more girls than boys prefer Logo programming, while in Year 7 significantly more boys than girls prefer programming to subject specific activities. One reason for this finding may be that Year 6 girls were the group with the highest intelligence in the sample. The more intelligent student may feel more challenged by Logo programming, not only because of its novelty.

Help Students in both years discuss computers and computing with peers and adults in and out of school. They ask for help from friends, parents, and other adults as well as their teachers.

Between April and November a 20% decrease was observed of Year 6 students discussing how they feel about computing with their friends; no difference in Year 7. Girls talk to their friends more than boys and Year 7 girls more than Year 6 girls. There was an increase of 20% for Year 6 and 30% for Year 7 between April and November of the number of students who discuss computing with members of their family. The increase for Year 6 girls is 30% and for Year 7 girls 43%. It is possible that due to school parent meetings, parents have become more interested in what their children do. Another possibility is that students are becoming desperate enough in their search for help that they are trying their families. The latter hypothesis finds some support in the fact that 22% more Year 7 girls, the weakest group in computing and the most alienated, are approaching their families. There is a 20% increase between April and November in Year 7 students overall seeking help from adults other than parents or teachers.

There was a decrease in Year 7 in the proportion of students whose friends help them with their computer work. There was difference in Year 6, and a decrease in Year 7 (not statistically significant), of those who help others in their computing. This may all be part of an increasing alienation.

Students' Expectations and Perceptions of Teachers In April 78% of Year 6 and 71% of Year 7 felt that at least sometimes *A computer is a bit like teacher.* In November, there was a 10-15% decrease in this belief for all groups. Are the students becoming realistic or disillusioned?.

Between April and November there was a 20% increase in the belief that computers can teach better than teachers for Year 6 and a 20% decrease for Year 7. At the end of the 1991 school year 32% of Year 6 and 16% of Year 7 students (28% for boys and 4% for girls) believed computers to teach better than teachers.

Students who use a personal computer need less attention from the teacher. Nearly 60% of the students agreed in April and November, the only change being that the stronger support for *often* in April became *sometimes* in November. However, 62% of Year 7 girls felt in April (but only 36% in November) this could never be so. These students realise that they are in frequent need of assistance from the teacher. In April 22% of Year 6 and 47% of Year 7 stated that it is difficult for teachers in the SUNRISE classroom to find the time to teach. In November 57% of Year 6 and 44% of Year 7 expressed this view, i.e. a significant increase in Year 6 and no significant change in Year 7.

In April 31% of Year 6 and 71% of Year 7 indicated that *Keeping computers, printers, etc. in working order takes a lot of the teachers' time.* The November data showed no significant change from this in Year 6 but a 20% decrease in Year 7. Does this mean that the teachers are now more effective in their maintenance of the technology? Is the technology in better shape? Or have these students adapted to the situation?

All teachers should know how to use and program computers. There is a stronger agreement with this statement for males in both classes and an increase (not statistically significant) between April and November in all groups. Most students believe that *Students can teach teachers things about computing.* There was a 17% increase in the proportion of Year 7 males and 3% increase in Year 7 females between April and November. The students are very aware of the fact that the teachers are not very far ahead of them in computing experience and that they are lacking much expertise. The weaker students appear to be becoming somewhat insecure. Most students enjoy teaching the teacher. But each class has about 10% of students who do not enjoy teaching the teacher. They may be just not good enough to do so.

Conclusion Students in both years have certainly improved in computer awareness and in the knowledge and skills relating to computing. In April expectations were high in both classes. During the year students became more realistic and critical of the SUNRISE offerings. By the end of the year a large number of the Year 7 students were unhappy about the prospect of another year, their Year 8 and first year at High School, in the project. Many of the students feel that they are missing out on certain learning and social activities which their friends in the non-SUNRISE class experience. It is difficult to judge how realistic these feelings are; what is clear is that they are widespread.

Surprisingly, the feelings and attitudes of Year 6 and Year 7 students are very similar, despite the fact that the Year 7 students have been in the project for 12 months longer than the Year 6 students. In Chapter 7 the feelings and attitudes of the students will be discussed in relation to ability and achievement.

STUDENTS' SELF-ASSESSMENT OF ATTITUDES AND COMPETENCE

A structured interview was devised to find out how the students themselves felt about computers and computing, and how they assessed their progress in computing in a number of subject areas. The interviews were carried out during the second term of the 1991 school year. Each student, individually, was asked the same set of questions in the same order by Irene Brown, who at this stage was well known by all students. Sometimes the interview took a few minutes, occasionally it led into other questions or explanations. The students were very cooperative and answered all questions readily. Each answer was recorded on audiotape, transcribed and coded. Results for each year level and gender were computed and compared. Some of the answers of Year 6 students reflect their more limited experience with computers.

The Interview Questions

The students were asked whether they used the computer *for fun* and if so exactly how they used it. They were also asked what they did on their computer at home. It was expected that responses to these questions would show whether students view the computer as something to be enjoyed and explored. With the SUNRISE classroom's emphasis on discovery learning and exploration, it could be postulated that students with a positive attitude towards the computer in terms of exploring and having fun would be learning more effectively. In asking the students what they do at home on the computer, insight can be gained as to whether they see the computer as a tool to develop their skills, a tool to help get required work done, an object to explore and learn from, or something with which to interact with others -- all or some of these. Answers also reflected variations in involvement in procedure writing. The children were then asked how they were getting on in different subject areas using the computer, as well as in typing and specifically word processing.

It is necessary to understand here the way the computer is used in the classroom. Computing is by no means an isolated subject; nor are typing skills and computing abilities presented as ends in themselves. The laptops are available to all students at all times, and are used as a tool to carry out schoolwork. Thus the computer is used in social studies for storing information, word processing and databases as well as graphics and animation. In mathematics it is used for calculations and to explore numbers and shapes. In language it is used as a word processor, with the addition of graphics and animation. The focus is seldom on computer programs for themselves. Even when a new procedure or concept is introduced, either by a teacher or a student, the emphasis is usually on the usefulness of that procedure. For example, the students are introduced to animation and then required to produce, over a period of weeks, an animated story. This time line allows exploration and practice with the new idea, but with a goal in mind and an end other than merely learning to program the computer.

The students were asked to explain how they knew that they were or were not doing well in the different subject areas, and from this was inferred their way of measuring success. *Are you doing well in mathematics on the computer?* was usually answered by a *yes* or *no*. The next question was then simply, *How do you know?* Answers reflecting the teacher's evaluation, other students' comments, or an individual's own assessment were given. Occasionally the student said he/she did not know, at which point they were prompted with *How could you tell if you were doing well or not?*

The children were asked to name a student whom they considered to be doing well in the particular subject area and why they thought he/she was doing well: *How do you know?* Some students were also asked how they thought the person they had named had come to do so well and whether they thought they could learn to do as well as that person.

Finally the students were asked a general question about what they thought was the most important reason for doing well in schoolwork and why learning was important.

Responses to Questions Relating to Computer Use

Question 1: *Do you use your computer for fun?*

	Yes %	Sometimes %	No %
Year 7 Boys	53	40	7
Year 7 Girls	62	31	7
All Year 7	58	36	7
Year 6 Boys	63	37	
Year 6 Girls	52	40	8
All Year 6	58	38	4
Total	58	37	5

The answer to this question showed very similar results for both year levels, and for boys and girls. Over half of the students responded *yes*, roughly 40% gave a qualified answer, such as *often* or *yes, sometimes*. The latter response for some may reflect more of a concern not to give an unconditional answer than showing any difference in attitude towards the computer. Only six students answered *no* to this question, and of these four are having difficulty in learning to write procedures, but the other two are successful by their own, as well as teacher's and peers' standards.

Question 2: *What do you do on your computer for fun?*

	Make up games %	Play games %	Graphics %	Writing %	Mazes %
Year 7 Boys	77	20	13	3	
Year 7 Girls	59	21	21	21	
All Year 7	68	20	17	12	
Year 6 Boys	40	30	23	7	33
Year 6 Girls	16	36	36	28	32
All Year 6	29	32	29	16	32
Total	49	26	23	14	

Despite the open-endedness of this question, the answers fell into the following broad categories: make up games, play games, graphics (i.e. picture or shapes), writing stories or letters. At the time of interviewing the Year 6 students had only just begun to learn how to make games; to voluntarily make games at this stage is certainly a reflection of their interest in exploring. They had also just started playing with and adapting mazes, so many of them chose this activity.

There were obvious gender differences in the answers to this question. More boys than girls made games in both classes, more girls than boys did writing and graphics. This confirmed the kind of gender differences that the teachers reported to have observed: *The boys do more exploring, while the girls take more time to present things nicely.* Implications of these gender differences are discussed in Chapter 8.

Question 3: *What do you do at home on your computer?*

	Homework %	Typing %	Shapes %	Pictures %	Programs %	Stories %	Games %
Year 7 Boys	87	67	77	50	87	67	77
Year 7 Girls	72	52	38	34	55	34	52
All Year 7	80	60	58	42	71	51	64
Year 6 Boys	70	80	20	27	7	10	23
Year 6 Girls	92	80	24	20	24	12	24
All Year 6	80	80	22	24	15	11	24
Total	80	69	40	33	44	32	45

Question 3 was asked to find out what students did when given total freedom in choosing how to use their computer. 80% of students in both classes reported that they are using their computers for homework set by the teacher. This obviously meets teacher expectations. 60% of Year 7 and 80% of Year 6 students used their computer at home for typing practice. Apart from these two almost compulsory activities, Year 7 students engaged in a far greater variety of computing activities at home than Year 6 students. For all activities mentioned, significantly more (13-50%) Year 7 boys than girls were engaged. For example, half the girls said they made games, whereas three quarters of the boys did. Of particular interest is that even for typing and story writing, activities which according to the research literature are generally preferred by girls, the Year 7 boys use the computer more at home than the girls. This finding further supports the view that the Year 7 girls in this sample are much less interested in computing than Year 7 boys and Year 6 girls and boys.

The picture was different for Year 6. When the interview was carried out with these students, programming was still quite new to them, so it is understandable that fewer activities were mentioned. About a quarter of Year 6, both boys and girls, said they were making games. About the same number were making shapes and pictures, and a smaller number were writing stories or letters. A quarter of the girls and a significantly smaller number of the boys were writing programs.

The finding that fewer Year 7 than Year 6 students practise their typing at home may be explained by the fact that by having been in the SUNRISE program for one extra year, the keyboard skills of Year 7 students might be more advanced than those of Year 6 students. Students are encouraged by their teachers to practise their typing at home. Year 6 students can be expected to be more compliant at this stage than Year 7, who altogether are less enthusiastic about their computing.

Responses to Questions Relating to Competence

The students were then asked to assess their competence in computing in several subject areas. This was done in order to gain an indication of the students' progress in these areas as well as to ascertain how they might assess their progress. The four areas explored were typing, procedure writing, mathematics and, for Year 6, stories, for Year 7, databases.

In the SUNRISE classrooms all students are encouraged to develop their keyboard skills and are provided with a typing tutor. Year 6 students are given time each day to practise typing, and students in all the classes are encouraged to practise at home and to keep a record of their progress on charts in the classroom.

Although procedure writing is not focused on as a subject in itself, it is part of all subject areas. Programming comprehension and efficiency certainly improves performance in all subject areas in the SUNRISE classrooms. Stories and databases were chosen as specific language areas to investigate, as much of the work in each is word processing. At the time of the interviews, each Year 6 student was involved in a language contract to make an animated story. This involved both writing a story and writing procedures for animation. When asked *Are you doing well in writing stories on the computer?* and *How do you know?* the answers sometimes showed that the students were thinking more about the animation than the language aspect. For example there were several answers like *Because it works*, or *It does what I want it to do.*

The Year 7 students did not have a specific language project at the time of their interview so it was decided to ask them about their work in making databases. This also involves some procedure writing but much of the work is in word processing. Again, responses showed that students were thinking about the procedure writing aspect of creating the databases.

The final area of investigation was mathematics. The students were asked *Are you doing well in mathematics with the computer?* Answers reflected that the students thought about mathematics skills as well as programming skills, both of which are important in completing work in mathematics in these classes.

Responses to the questions were divided into four alternatives: Yes (with no qualification), Yes qualified (including responses such as *yes, OK* or *most of the time*), OK and No.

Question 4: *Are you doing well in typing?*

	Yes %	Yes, qualified %	OK %	No %
Year 7 Boys	43	20	17	20
Year 7 Girls	65	10	17	7
All Year 7	54	15	17	14
Year 6 Boys	63	17	7	13
Year 6 Girls	68	16	16	
All Year 6	65	16	11	7
Total	60	16	14	11

Over half the students gave an unqualified yes to this question, and three-quarters of responses were definitely positive. The answers did reflect gender differences at both year levels, but more so in Year 7. Fewer Year 7 boys gave a qualified yes to the question, and more boys said no at both year levels. There were no Year 6 girls who said they were not doing well at typing.

Question 5: *Are you doing well in writing procedures?*

	Yes %	Yes, qualified %	OK %	No %	Don't know %
Year 7 Boys	56	30	7	7	
Year 7 Girls	34	10	31	24	
All Year 7	46	20	19	15	
Year 6 Boys	46	10	27	13	3
Year 6 Girls	28	40	24	8	
All Year 6	38	24	25	11	2
Total	42	22	22	13	1

In the total sample fewer than half of the students were confident enough to give an unqualified positive answer to this question. Girls in both Year 7 and Year 6 were much less likely to say yes, and in Year 7 much more likely to say no, a quarter of the Year 7 girls doing so.

Question 6: *Are you doing well in making databases/stories?*

	Yes %	Yes, qualified %	OK %	No %	Don't know %
Year 7 Boys	77	7	17		
Year 7 Girls	75	7	7	10	
All Year 7	76	7	12	5	
Year 6 Boys	53	20	20		7
Year 6 Girls	64	16	8	4	8
All Year 6	58	18	15	2	7
Total	68	12	13	4	4

Three-quarters of the Year 7 and more than half of the Year 6 students responded to this question with an unqualified yes, and over half of the Year 6 students did. In general both boys and girls, at both levels, were confident about their competence in creating databases or stories, although several students, all girls, felt that they are not doing well in making databases/stories. These results reflect the generally positive attitude towards word processing. When asked to identify *the best thing about using computers* many students commented on the fact that they did not have to use paper and pencil, and that erasing was easy.

Question 7: *Are you doing well in mathematics with the computer?*

	Yes %	Yes, qualified %	OK %	No %	Don't know %
Year 7 Boys	43	17	23	13	3
Year 7 Girls	44	7	24	24	
All Year 7	44	12	24	19	2
Year Six Boys	70	13	13	3	
Year 6 Girls	60	16	20		4
All Year 6	65	15	16	2	2
Total	54	13	20	11	2

Year 6 students felt much more confident about doing well in mathematics than Year 7. On the whole it appeared that the Year 6 students were enjoying using the computer in mathematics, and were finding this subject easier than in previous years. For the Year 7 students both difficulty in understanding mathematical processes and difficulty in writing procedures hindered them. A number of Year 7 girls tended to classify themselves, *I'm just not good at mathematics/writing procedures*. One-fifth of Year 7 students (25% of the girls) said they were not good at mathematics on the computer.

In a comparison of the results of both year levels, boys and girls, (see Figure 6.4) Year 6 students are shown to be much more confident in mathematics and typing but less confident in the other areas. Girls are a lot more confident than boys in typing and much the same in the language area of databases and stories. However, boys are more confident in procedure writing and mathematics. The area of least confidence is procedure writing, and generally most confidence is shown in databases/stories.

The groups showing the most confidence in each subject area are the girls in typing, the boys in procedure writing, Year 7 students as a whole in databases, and Year 6 students as a whole in mathematics.

Figure 6.2 shows the answers to the question 'Are you doing well in . . .' across the subject areas and gives interesting comparisons. The Year 7 boys, in all subjects apart from typing, are more likely than the girls to claim they are doing well. This was particularly marked in procedure writing, where the girls were least confident. Mathematics is the area of generally least confidence and databases of most confidence.

Figure 6.3 shows smaller differences across the subject areas for Year 6 than for Year 7 students. Procedure writing was the area of least confidence. Girls, particularly, were unwilling to say without qualification that they were doing well in this area.

How the Students Measured Success

A question asked of students with respect to each subject area was *How do you know you are doing well?* The responses provide some indication of how the students measured success, and how this measuring differs according to the subject area.

In *typing* over half the students at both year levels based their self-evaluation on the results shown by the typing chart. Their achievement with respect to typing speed, accuracy, figures and graphs served as measures of success. The Year 6 girls particularly were aware of their exact typing speed. The Year 7 students also compared themselves with others and with how well they had done previously. A number of the Year 6 students said they did not know how to tell how well they were doing.

The criteria for success in *procedure writing* were rather different. By far the most common criterion for success was simply whether or not the procedures work. This was referred to by two-thirds of the Year 6 students but not quite one-third of the Year 7 students. At this stage of their development, whether procedures work or not would probably be the only available measure for Year 6 students, whereas the Year 7 students could be much more aware of the quality of a procedure. These students, therefore, used measures such as improvement, comparison with others, their own understanding, and others' comments or help.

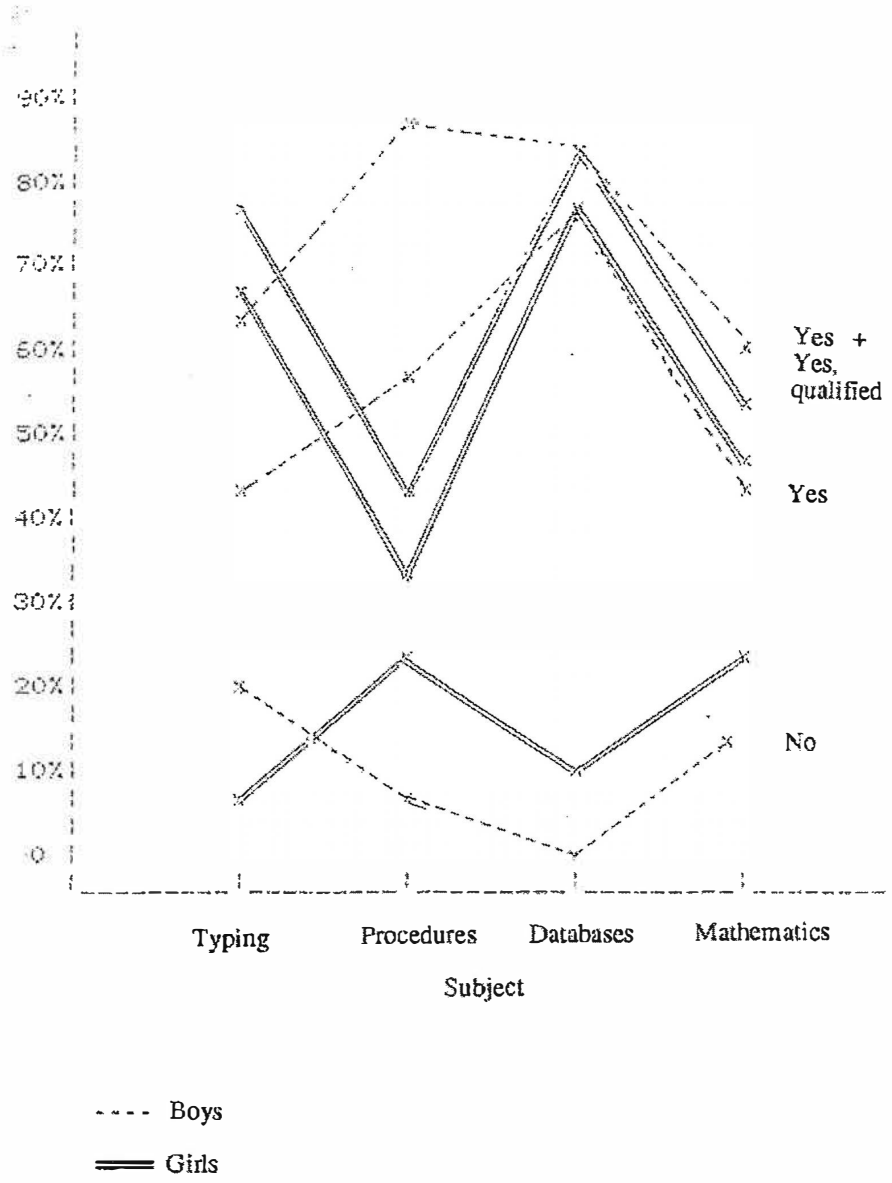


Figure 6.2 Results of Year 7 Self-assessment of Competence

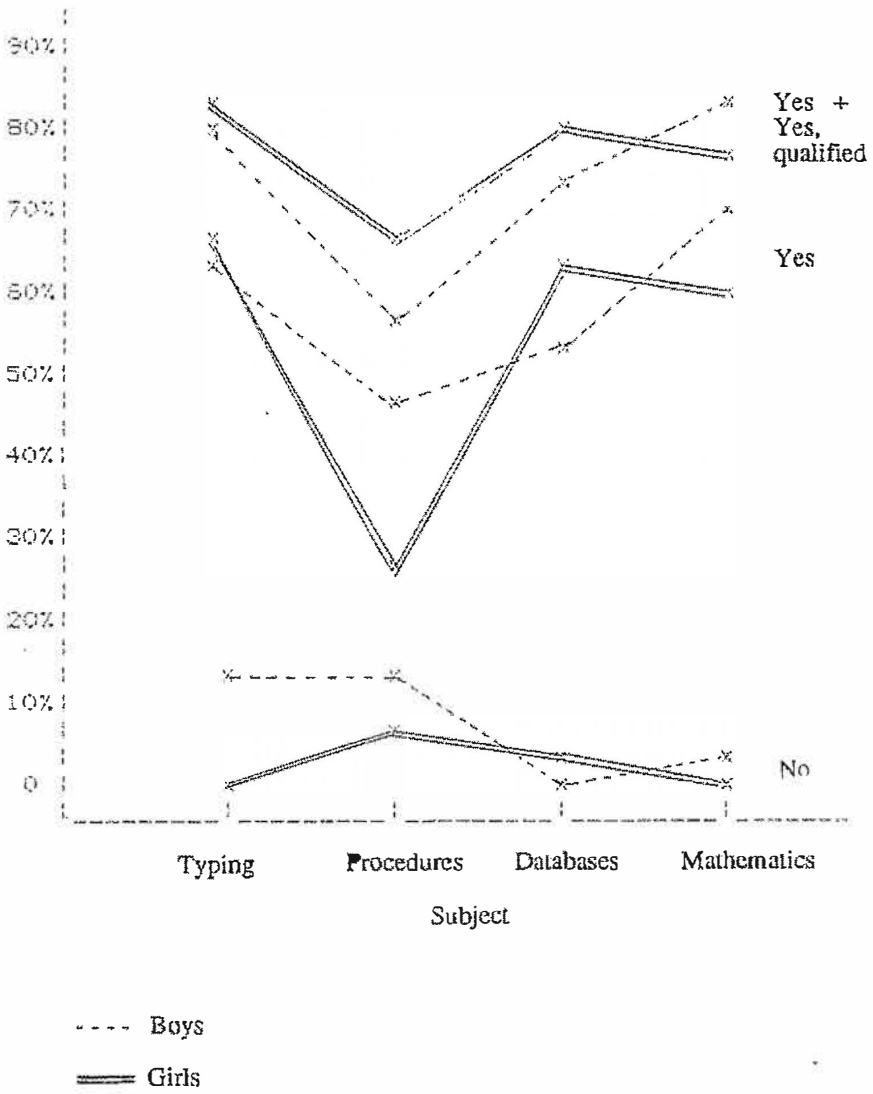


Figure 6.3 Results of Year 6 Self-assessment of Competence

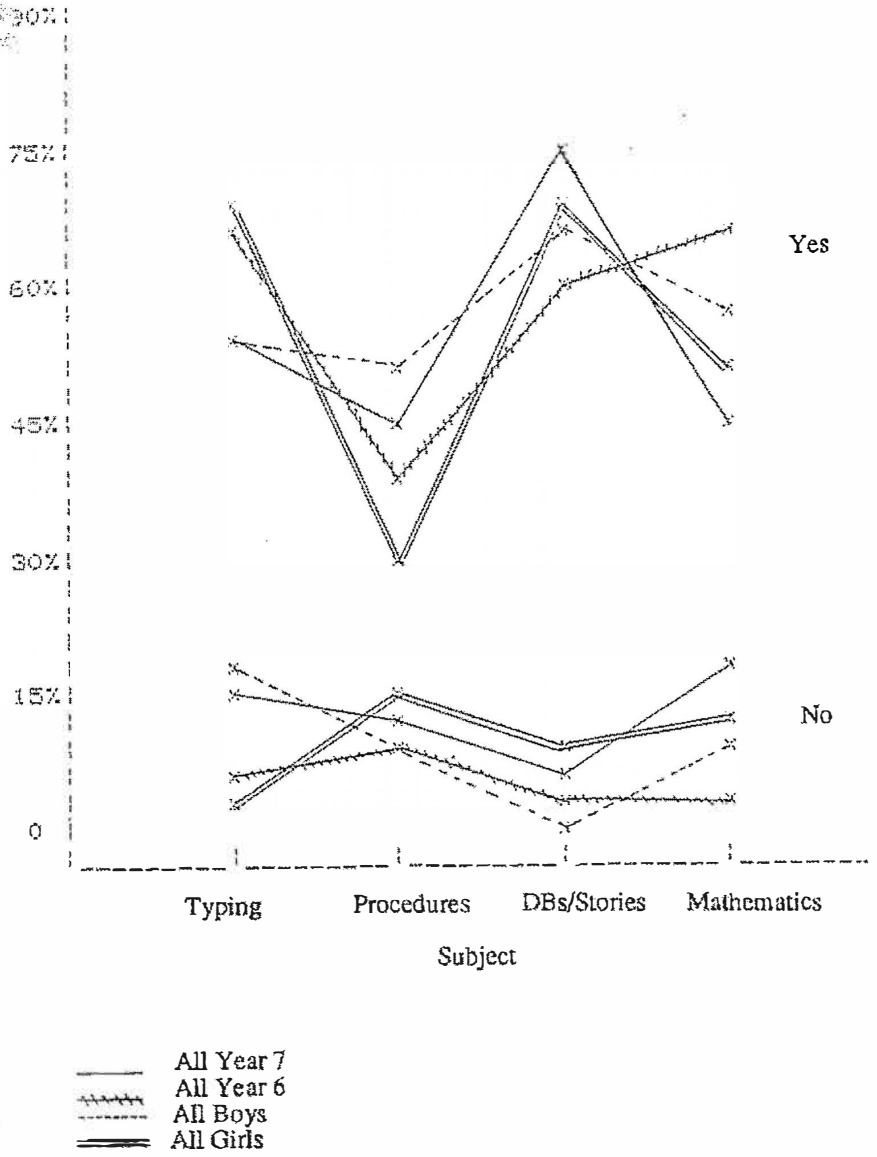


Figure 6.4 Self-assessment of Competence by the Total Sample

Both Year 6 and Year 7 girls (18%) used subjective measures, such as their *ability to understand*, more than the boys did (7%).

The third area investigated, *databases* or *story-writing*, showed different measures for different groups. 60% of the Year 7 girls commented on whether the procedures worked -- whether or not they could make a database. One-third of the boys at that level used teacher gradings as a measure of success. At the Year 6 level one-quarter of both boys and girls depended on what peers said to know how well they were doing, and 15% did not know how to tell. It should be noted that their animated story project had not yet been marked by the teacher so they did not have access to this type of feedback. More than one-third of all the students relied on some specific aspect of the task to measure success, e.g. they had a lot of information, their animation was good, and so on.

Mathematics was another subject area in which the students could use immediate results to measure their success. Being able to do the work or write the procedures accounted for over one-third of the Year 7 students' answers, whereas one-fifth of Year 6 students referred to whether their procedures worked. A quarter of the Year 6 boys relied on getting the answers *right*. Apart from these, the Year 7 students compared themselves with others or their own previous success. Year 7 boys relied on grades given by teachers, and the girls at both levels on their *ability to understand*. Again, there were a number of Year 6 students who did not know how to tell how well they were progressing.

Table 6.4 shows the results across the four subject areas. Year 7 students compare themselves with others and with previous ability, whereas the Year 6 students seldom refer to these measures. Year 7 boys refer to outside measures, such as grades received from teachers, far more than any other group of students, and girls at both levels use a subjective measure, such as their understanding, more than the boys do. Obviously, Year 6 students are still learning how to measure success, more than Year 7. In each subject area an average of 10% said they did not know how to tell if they were doing well or not.

Identification of Competence in Others

In a third aspect of the questions relating to specific subject areas, the students were asked to name a peer whom they judged to be doing particularly well. Response differences related to the varying number of people named, gender differences, and the tendency to name the recognised *class expert*.

In all subject areas there was a tendency to name someone of the same sex. Thus if someone of the opposite sex was named it could be assumed that he/she was doing particularly well. From this it is evident that girls are seen as doing much better in typing. Over 60% in both years named a girl as being good, over 80% of the girls named a girl and 40% of the boys named a girl in both classes. This was the only subject area in which this happened. In the other three areas

Table 6.4 Student Measures of Success

	Typing		Procedures		Stories/DB		Mathematics	
	Y7 %	Y6 %	Y7 %	Y6 %	Y7 %	Y6 %	Y7 %	Y6 %
Objective e.g. marks, report	58	51	3	2	19	4	14	5
I am improving	11	-	12	2	-	-	7	2
Better than others	7	-	10	-	7	-	7	-
Subjective e.g. I understand	-	-	14	12	-	-	14	11
Others say so	-	7	10	4	13	25	-	-
Others help me	-	-	12	2	-	-	-	-
I can/can't do it	8	6	29	67	37	6	39	33
Some specific aspect works	-	-	-	-	10	27	-	-
Don't know	-	11	-	4	3	15	-	13
Other	11	11	-	9	12	15	-	15

50% - 70% named someone of the same sex. In mathematics, however, a third of the girls named a boy as being particularly good.

Table 6.5 shows how frequently a boy or a girl was nominated as an *expert* in the four subject areas. There were differences between the year levels in the nominations of girls or boys, the exception being in typing where girls were nominated by the students in both years. Year 7 consistently favoured boys in the other three subject areas, especially in writing procedures, where 80% of the Year 7 students nominated a boy. Year 6 favoured girls in these three areas although in mathematics the proportions were fairly close.

These results reflect students' perception with respect to class *experts*. As an instructional method SUNRISE teachers have devised a scheme called the experts scheme whereby the teachers will work with a small group of students on a new procedure or concept. These students then become the *experts* in that procedure and become a resource for the other students. One boy in Year 7 consistently writes superior programs and is recognised by the teachers and students as the class expert. In the Year 6 classes there is a girl who had been recognised in previous years as a quick learner. This reputation earned her a place among the experts in early procedure writing, and was subsequently recognised by many students as the Year 6 *expert*.

In the naming of those doing well in writing procedures the same Year 7 expert was named by over 60% of both boys and girls. The Year 6 expert was named by 60% of the girls and 25% of the boys. These students gained the most marked support, but the two experts were nominated in other subject areas as well. In database writing, stories, mathematics and typing about a quarter of the students in each class named these experts. An exception was found in mathematics, where only 13% of the Year 6 boys nominated this particular girl,

Table 6.5 Percentage of Students who named a Boy *Expert* or a Girl *Expert* in four Subject Areas

	Typing		Procedures		Stories/DB		Mathematics	
	Boy %	Girl %	Boy %	Girl %	Boy %	Girl %	Boy %	Girl %
Year 7	24	63	80	17	53	32	69	36
Year 6	13	67	20	51	18	36	18	26
Boys	33	43	65	14	55	5	55	12
Girls	1	89	33	72	14	62	32	53

and in typing, where the same percentage of Year 7 students named the class expert while 60% of them nominated two girls instead.

The number of different students nominated varied according to the subject area. In typing only one-fifth of the students at each year level were nominated. In all the other subject areas one-third of the students were nominated. The criteria for doing well in typing appear to be clearer and also more public (there are wall charts of progress). Less than half of the students at each year level were nominated more than once, while 30% of Year 7 and 20% of Year 6 were nominated more than three times. The results of these nominations coincided in most instances with the teachers' assessments of which students were progressing most successfully.

The Year 7 students appeared to be much more aware of those doing well in each area. Only an average of 5% said they did not know whom to nominate, but 23% of the Year 6 students gave a *don't know* response. This was particularly obvious in story writing where nearly half of the Year 6 students did not know whom to nominate.

Agreement between Self-assessment and Peer Nominations

As each person had been asked earlier in the interview how well they thought they themselves were doing in each subject area, it was possible to check whether the students' self-assessment agreed with the peer nominations. Of those nominated as doing particularly well, 92% of the Year 7 students and 80% of the Year 6 students assessed themselves as doing well, or gave a qualified Yes answer.

The final question asked in each subject area was how the student knew that the person they had nominated was doing well. Table 6.6 shows the criteria the students used in measuring success in their peers. (If students said they did not know who was doing well they were not usually asked how they would know who was doing well.)

Measures of Success of Others

	Typing		Procedures		Stories/DB		Mathematics	
	Y7 %	Y6 %	Y7 %	Y6 %	Y7 %	Y6 %	Y7 %	Y6 %
Objective e.g. marks	22	42	-	-	-	-	-	-
I work with them/ see them	31	24	13	18	26	27	24	7
They're good/ they know	15	15	8	4	7	4	-	3
Does good work	-	-	44	31	14	2	8	10
Teacher's comment	10	4	-	-	-	-	14	-
They help me	-	-	14	16	17	2	16	11
Some specific aspect	8	-	12	16	25	15	17	5
Other	10	4	-	7	7	11	9	17

Criteria for Judgment

Typing appears to be the only subject area with clear objective criteria by which the students can measure their own progress and that of others. Most measures of the competence of peers were subjective, e.g. *I work with them; I see their work, they know what to do; They are good.* Girls used this last judgment twice as often as the boys did. *Does good work* was fairly general, as were *They write good procedures; They make good databases*, and in mathematics *They get the answers right.*

It appears that at the stage of interviewing the Year 7 students relied more on teachers' grades and judgments than the Year 6 students did. The students who relied on teachers' judgments commented on the teachers asking students for their results or answers, or the teacher commenting on someone's work. Many of the students judged how good others were by how much they could help them when they experienced difficulties. With the emphasis in the SUNRISE classroom on cooperative learning, the students learn a lot from each other, and ask each other for help. It appeared to be difficult for both Year 6 and Year 7 students to judge the success of others in mathematics. Responses were varied, and students tended to feel that there are no clear ways of knowing how someone else is getting on.

Some differences could be observed between boys and girls and the two year levels. For example, Year 7 girls were much less aware of speed and accuracy results in typing than the other students.

Most of the Year 7 students were asked how they thought the peers they had nominated came to be so good at typing and at writing procedures. For typing almost all of them answered that it was by practice. When asked if they thought

that they personally could become as good at typing as their nominated peer the girls were much more confident than the boys -- only 45% of the boys compared with 65% of the girls felt they could become that good at typing. This may be due to the fact that the top typists in the class are girls. The response was different for procedure writing. Success here was attributed to learning from others, from the teachers, from the manual and by practising. Less than one-third of the students thought that they could be as competent as the person they had nominated. Again there was a suggestion in the responses that if the person nominated was of the same gender, the student who nominated him/her was more confident that he/she could also succeed.

General Questions

Question 8a: *What is the most important reason for doing well in your schoolwork?*

	Job %	Marks report %	Edu-cation %	Pass grade %	Learn %	Achieve %	Be better Be smart %	Uni %	Don't know %
7 B	63	17	7	3	7	3	-	-	-
7 G	62	10	10	-	7	3	5	-	-
7	63	14	8	2	7	3	3	-	-
6 B	40	13	13	-	10	3	-	3	3
6 G	40	24	4	8	16	-	-	8	-
6	40	18	9	4	13	2	-	5	2
6&7	52	16	9	3	10	3	2	3	5

Question 8b: *Why is it important to learn things?*

	Job %	Teach other %	Marks %	Good at things %	Pass grade %	Know specifics %	Know things %	Be smart %	Don't know %	Other %
7 B	33	7	-	7	-	10	10	10	3	13
7 G	38	7	3	-	3	10	10	3	10	7
7	35	7	2	3	2	10	10	7	7	10
6 B	57	5	5	5	-	10	10	5	-	5
6 G	29	5	5	5	-	14	28	-	-	14
6	43	5	5	5	-	12	19	2	-	9
6&7	39	6	3	4	1	11	14	5	4	10

The interview finished with a general question about schoolwork and learning. The students were asked what they thought was the most important reason for doing well in schoolwork and why learning was important. The most frequent single reason given was *for a job*. More than 60% of the Year 7 students and 40% of Year 6 gave this response. Other reasons given by more than one-eighth of the students were *for good marks* or *to receive a good report* and in order *to learn*. Other reasons were *to get an education; to pass the grade; to achieve; to be smart* and *to be a better person*.

More varied answers were given to question 8b than 8a. In response to Question 8b a little over one-third of the students said *for a job*, but other responses referred to teaching other people (including children), being good at things, knowing things, including specific examples such as reading and using money. Again, there were references to marks, passing grades, and being smart.

CONCLUSION

This interview provided rich insight into how the students themselves used the computer and how they viewed their progress in computing. The questions concerning how much the students use the computer for fun showed a majority of students enjoying computing and exploring with it. Almost all of the students use the computer for fun at least *sometimes*. While 80% of the students used the computer for homework and typing practice at home, a lower proportion engaged in exploration. Most of the Year 7 boys were involved in making games at home, while fewer girls were involved in this sort of activity. At the stage of interviewing the Year 6 students seemed to be less able to engage in such activities as making games and programming. The much higher confidence in programming shown by the boys may be related to their higher involvement in exploratory activities at home.

Students were asked to comment on how well they were doing in computing in each of four subject areas. Answers here differed according to class level and gender. The Year 7 students as a whole were most confident in making databases and least confident in mathematics. At both year levels the girls were more confident than the boys in typing but the boys felt that they excelled in procedure writing. The Year 6 students showed less variability across the subject areas, with procedure writing being the area of least confidence. They were much more confident in mathematics than Year 7, but less confident in the language area. Performance on standardised tests of educational achievement (without using computers) had shown that achievement in language of Year 7 was relatively lower than the achievement of the students in Year 6. With and without computers the Year 7 students probably feel less confident in language skills as they realistically assess their abilities.

In asking how the students knew that they or others were doing well, criteria for success were sought. Again, responses differed according to the subject areas.

Typing was the area in which students seemed most clear of their progress. Certainly typing is the activity in which the most immediate and consistent feedback is provided. Suggested measures of success in the other areas varied from external measures such as teacher assessment and marks, to feedback (comments) obtained from others, i.e. both peers and teachers, to quite subjective measures, such as *I just think I'm good, I understand it*, or comparative judgments, e.g. *I'm better than other people; I'm better than I used to be.*

The students were asked to name a peer whom they considered to be doing well in each subject area. Although some named a friend rather than an expert, overall the people named consistently were recognised also by the teachers as those students who were most successful. Students at both year levels tended to name a peer of the same sex, although the Year 6 students, in contrast to Year 7, recognised a girl as the class expert. In Year 6 girls were nominated more frequently than they were at the Year 7 level. An important finding was that if the person nominated as successful was of the same sex the student was more likely to be confident of their own possible success in that area.

The students nominated most often shared certain characteristics. While, as a group, they were not necessarily more successful than the other students in typing and mathematics, they were seen to be very successful in procedure writing, and databases or story writing. As a group they also used the computer for fun especially for making games in their spare time.

From these results it appears that the key area for judging success in oneself or other students is procedure writing, i.e. programming. While typing gave some confidence and fluency, and mathematics and language skills were important in certain subject areas, it was procedure writing which appeared to be the main criterion for the perception of overall expertise.

Students' self-assessment in the area of procedure writing was compared with their performance on several programming efficiency tasks. Of those students who performed well in these tasks (the top quarter of each level) two-thirds had said that they were doing well in programming. Apparently they perceive their performance accurately. However, of those who performed poorly, about half (a little under for the Year 7 students) also said they were doing well, or reasonably well. A similar pattern was found when students' self-assessment was compared with the teachers' assessment of them. Students whom the teachers perceived as doing well saw themselves as succeeding, while a proportion of those seen by the teachers as performing poorly perceived their own performance as quite acceptable. It appears then, that those who can perform well perceive correctly that they are doing so, but other factors are taken into account by those who are performing poorly. This is not unexpected when one understands the kinds of criteria the students are using to judge their own success. For example, students who perform poorly, but can see that their performance or understanding has improved since last year understandably comment that they are doing well.

7

Individual Differences in Attitudes and Learning

This chapter shows how individual differences such as ability, measured intelligence, and competency in computing can relate to students' feelings and attitudes about educational computing, knowledge, self-concept, expectations and perceptions of teachers, etc. The second part of the chapter discusses findings relating to the students' demonstration of their knowledge and understanding of computer programming. The chapter concludes with a brief demonstration of the interaction between students who were engaged in a collaborative programming assignment.

ABILITY AND INTELLIGENCE

The literature suggests that intelligence and general ability strongly influence achievement in computing as they influence learning in other educational areas. The following section summarises the intelligence test performance of the sample and reports on the feelings, attitudes and perceptions of the students in different IQ classifications.

For these analyses students were classified into 3 groups according to their IQ. The *Above Average* category contained the students whose IQ was 110 or higher. The *Average* category covered the IQ range 90 - 109, and the *Below Average* category contained the students whose IQ was 89 or lower. The three categories used here are based on the traditionally and widely used intelligence classification, which was initially devised by David Wechsler. The *Average* category covers the same range as Wechsler's. Because of the small sample size, the top and bottom categories used by Wechsler were amalgamated for the purposes of the present analysis. Table 7.1 shows the mean IQ and standard deviations for the total sample and various subgroups. Table 7.2 reflects the proportions of students in the sample in different IQ categories.

Regardless of year level the girls tended to obtain higher IQ scores than the boys. In the total sample the average IQ for girls was 107 (SD 11) and for boys 102 (SD 12, $p<.05$). Year 6 girls showed the highest average IQ, i.e. 108 (SD 12), followed by 106 (SD 11) for Year 7 girls, then Year 6 boys with 104 (SD 12) and

Table 7.1 IQ Levels of the Sample

	N	Mean	Standard Deviation
Total sample	115	104.13	12.13
Year 6	56	105.75	12.17
Year 7	59	102.59	12.00
Girls	55	106.51	11.46
Boys	60	101.95	12.41
Year 6 girls	26	107.54	11.90
Year 6 boys	30	104.20	12.38
Year 7 girls	29	105.59	11.17
Year 7 boys	30	99.70	12.24

Year 7 boys with 100 (SD 12). None of these differences are significant. The differences between mean IQs for gender within Year were not statistically significant ($p>.05$).

The percentage figures in brackets below the intelligence categories indicate the proportions expected to occupy these categories under the theoretical normal curve.

Feelings and Attitudes of Students in Different IQ Categories

Anxiety As was reported previously, understandably, all students were more anxious that something might happen to their computers, and about computing more generally, in April than in November. In April about a quarter of the students of Above Average and Average IQ but 43% of those in the Below Average IQ category reported to still be feeling nervous when they are using their computers. In November only 6% in the Above Average IQ group and 17% in the Average and Below Average groups reported feeling nervous. In November 63% (April: 58%) in the Above Average, 81% (April: 40%) in the Average and 77% (April: 64%) in the Below Average groups stated *Computers don't scare me at all*. There was certainly a highly significant reduction of anxiety among the students in the Average range and less anxiety in the Below Average range of general ability. The specific fear of physically damaging the computer was reduced by 31% for Above Average, 21% for Average and 41% for Below Average students between April and November. In November 21% of Above Average, 25% of Average and 23% of Below Average students are still frightened that they might damage their computers.

Table 7.2 Proportion of Students in Different Intelligence Classifications

		IQ											
		≥130 (2.2%)		120-129 (6.7%)		110-119 (16.1%)		90-109 (50%)		80-89 (16.1%)		70-79 (6.7%)		≤69 (2.2%)
Sample	N	n	%	n	%	n	%	n	%	n	%	n	%	
Total	115	3	3	8	7	24	21	66	57	12	10	2	2	0
Girls	55	3	5	3	5	13	24	32	58	4	7	0		0
Boys	60	0		5	8	11	18	57	34	8	13	2	3	0
Year 6	56	1	2	6	11	14	25	29	52	6	11	0		0
Year 7	59	2	3	2	3	10	17	37	63	6	10	2	3	0
Year 6 Girls	26	1	4	2	8	7	27	14	54	2	8	0		0
Year 6 Boys	30	0		4	13	7	23	15	50	4	13	0		0
Year 7 Girls	29	2	7	1	3	6	21	18	62	2	7	0		0
Year 7 Boys	30	0		1	3	4	13	19	63	4	13	2	7	0

		Above Average (25%)		Average (50%)		Below Average (25%)	
	N	n	%	n	%	n	%
Total	115	35	30	66	57	14	12
Girls	55	19	35	32	58	4	7
Boys	60	6	27	34	57	10	17
Year 6	56	21	38	29	52	6	11
Year 7	59	14	24	37	63	8	14
Year 6 Girls	26	10	38	14	54	2	8
Year 6 Boys	30	11	37	15	50	4	13
Year 7 Girls	29	9	31	18	62	2	7
Year 7 Boys	30	5	17	19	63	6	20

Enthusiasm Our data show no significant differences with respect to the enjoyment students experience in using their computers between April and November and across IQ levels; nor was there any difference in the curiosity of students about *how the computer works*. Over 80% of the students felt high enjoyment and curiosity. However, 11% of students in the Above Average, 25% in the Average and 46% in the Below Average range still cannot understand how some people enjoy spending so much time on computing.

In November 13% more of the students in the Below Average IQ range than in April, and thus the same proportion as in the Above Average and Average range, regard their laptop as their *friend and helper*. The proportion of students who report that they are finding it hard to stop working on their computers once they

have started dropped by 26% for the Above Average, 14% for the Average and 18% for the Below Average groups between April and November.

The proportion of students who stated in November that they would *not be unhappy to be in a class without computers* was 10% higher than in April. 26% of Above Average, 20% of Average and 39% of Below Average students still feel that they would not be unhappy in a class without computers. This view may result from different reasons for the three groups. Above Average and Average students may have realistically assessed that they would be able to function equally well in a class without computers. Other observations suggest that the Below Average and some of the Average students may actually prefer a class without computers as they are finding it difficult to cope with the programming aspects of computing.

In all IQ categories only about half of the students looked at books about computing outside the classroom in November, i.e. 25% fewer than in April. Over 60% of the students never try to find books about computing in the library. The latter figure has not changed significantly from April. This state of affairs might suggest that the students are gaining sufficient information in the classroom; it might also be an indication of a weakening in interest in computing over time. The latter explanation finds support in the finding that in the Above Average and Below Average groups interest in *figuring out what I did wrong in my computing* was reduced by close to 50% and in the Average group by 14% between April and November.

Self-image and Confidence Although not statistically significant, a higher proportion of students in the Average range than in the Above Average and Below Average groups believe strongly that when they are working with their computers they will do particularly well in their work. It is likely that the students of Above Average and Below Average intelligence have learnt about the limitations resulting from insufficient skills on the part of the computer operator. They may also be more realistic about the technology than their peers in the Average group. A considerable number of the students in the latter group understood *doing well* as relating to typing and typed presentation of work, rather than to more general aspects of the curriculum.

More students in the Above Average range than in the other two groups believe that they will do better and better in computing, with the Below Average students being most doubtful about their eventual improvement. In November only 6% (April: 12%) of Above Average, but 33% (April: 46%) of Average and 39% (April: 22%) of Below Average students felt that they are *not the type to do well in computing*. However, a significantly lower proportion of Below Average students than students in the other groups report *getting annoyed* with their computers. This again shows the realism on the part of the Below Average group, who obviously appeared to have had higher initial expectations of computing than experience bore out. Very few students in the Above Average and Average groups, but 23% in the Below Average group believe that some students in their class are finding computing even more difficult than they themselves do.

In November 66% (April: 53%) of Above Average, 56% (April: 39%) of Average and 85% (April: 55%) of Below Average students discussed how they feel about their computing with their friends. Significantly fewer students in the Above Average range (and 30% fewer than in April) than in the other groups discuss computers and computing with members of their family, while approximately half of the students in all groups talk about their computing with adults other than parents and teachers. For the Below Average students this constitutes a 23% decrease between April and November. It is likely that these students have fallen behind their peers so much that they are lacking the confidence required to discuss computers and computing with adults.

Knowledge of Computers and Computing Between April and November the proportion of students who felt that using the computer is time-saving fell by 17% in the Above Average and 12% in the Average groups, but increased by 25% among the Below Average students. During the same period agreement with the statement *Learning to work with computers is just as important as reading, mathematics and spelling* fell by 10% in all three IQ groups, while support for the statement *Every school child and adult should be able to use personal computers* fell by 20% in the two upper IQ groups and by 40% in the Below Average group. It would appear that as computing is becoming more difficult (perhaps because the relevant skills at this level are largely cumulative and it is thus easier to *fall behind*), students are rationalising their positions. Yet all students believe that computers are becoming increasingly important in people's lives and will be of utmost importance to our society in the future; and 89% of Above Average, 84% of Average and 77% of Below Average students agree that computers have improved people's lives.

However, fewer students in the Below Average range (and with a further 10% reduction between April and November) than in the other groups would like to know how the computer works.

Learning and Problem Solving Between April and November there was a 20% increase in the proportion of students in the Above Average range and no change in the other groups in the feeling that when working with computers they will get their work done well. But there was a 21% reduction in the Above Average, a 10% increase in the Below Average and no change in the Average group in the view that the laptops allow students to work independently.

In November 69% (April: 59%) of Above Average, 47% (April: 18%) of Average and 62% (April: 59%) of Below Average students reported that computing often or sometimes makes schoolwork more difficult. A total of three students in the Above Average and Average groups but a quarter of the Below Average students denied that computing is at least sometimes *hard work*.

In November the number of students in the Above Average range who found that *computing makes learning in mathematics easier* was 20% lower than in April. There was no change in the other IQ groups. In November 31% of Above Average, 16% of Average and 24% of Below Average students believed that computing does not make it easier to learn mathematics. Judging more generally, 20% of Above Average, 14% of Average and 17% of Below Average students do not believe that computers make a difference to learning, i.e. that children who use computers do better in their school work. In November 29% (April: 49%) of Above Average, 64% of Average and 69% of Below Average students believed that *computers make more mistakes than clever humans*. There was no difference in this view between April and November for the latter two groups.

More than 50% of the students in the Above Average and Average groups are finding computing most useful in mathematics, while 31% of the Below Average group hold this view. 23% of Above Average, 17% of Average and 31% of Below Average students are finding the computer most useful in social studies, and only 17% of Above Average, 8% of Average and none of the Below Average students chose writing (i.e. word processing) as the area in which they are finding the computer most useful. Other subject areas were mentioned by individuals only. A mere 2, 10 and 3 students in the Above Average, Average and Below Average groups respectively are finding the computer useful in all their curriculum subjects.

A significantly larger proportion of Below Average than Average and Above Average students believed in April that the computer is a just like a hammer or lathe. In November 83% of Above Average, 88% of Average but only 75% of Below Average students took this view.

Control As noted above, between April and November there was a 21% reduction in the Above Average, a 10% increase in the Below Average and no change in Average groups in the feeling that the laptops allow students to work independently. 88% of Above Average, 83% of Average but only 69% of Below Average students believe that working with their computer puts them in charge of their own work. The latter proportions represent a 10% increase in this feeling of control since April for the Above Average and Average groups, but a slight decrease for the students in the Below Average range. 87% of Above Average, 88% of Average and 92% of Below Average students believe that they have control over what they do when they are using a computer. For the Below Average group this constitutes a 35% increase in the feeling of control compared with the data collected in April. Similarly, only 9% of Above Average and 6% of Average, but 31% (an increase of 24% from April) of Below Average students feel that they are *never* the boss of their computer.

Favourite Activities Using Computers Students were asked to name their favourite activities in computing. The responses were classified into two types: computer programming related activities and activities relating to specific subject

content. In November 62% (April: 78%) of Above Average, 49% (April: 71%) of Average and 67% (April: 60%) of Below Average students named programming related activities. In November 18% of Above Average and 12% of Average students, but none of the Below Average students, named playing games. This latter activity was not mentioned by any of the students in April.

When asked *What are the three best things about using laptops?* in November 74% (April: 91%) of the responses of Above Average, 81% (April: 82%) of the Average and 77% (April: 93%) of the Below Average students were related to learning, knowledge and skill development; 49% (April: 37%) of the responses of Above Average, and 31% (April: 29%) of the responses in the two other groups related to the more pleasing physical presentation of work done on the computer; 31% (April: 11%) of Above Average, 20% (April: 17%) of Average and 23% (April: 7%) of the Below Average students stressed the time-saving and other convenience aspects of computing, and 40% of all students noted that computing is fun (without qualification). Only 21% (April: 9%) of the Above Average and 2% (April: 9%) and 8% (April: 7%) of Average and Below Average students respectively recognised computing as an asset for the future, increasing job opportunities, etc.

When asked in which subject area they enjoy using the computer most, 49% (April: 12%) of the Above Average, 33% (April: 13%) of Average and 39% (April:29%) of Below Average students named social studies, next came language, then word processing. Only 3 students in the Above Average, 5 students in the Average and 2 students in the Below Average groups named mathematics. This latter finding is interesting when one remembers that more than half of the students in the Above Average and Average groups and 31% of those in the Below Average group believe that the computer is most useful in mathematics. Recognising the usefulness of the tool is obviously not the same as liking the work to be done with it.

In November only 6% (April: 41%) of Above Average, 9% (April: 20%) of Average and 15% (same as in April) of Below Average students stated that they enjoyed using the computer in all subject areas. Two students in the Average group said they did not like using the computer in any subject area. There is an obvious need for teachers to make students more aware of the advantages of computing in acquiring subject related knowledge and skills. No significant differences were found between the IQ groups with respect to the subjects students report to be liking less now that they are working with computers. Science is the subject nominated as being liked less by the largest proportion of students at all IQ levels, and next comes mathematics.

Help As noted previously, students in the SUNRISE classes are encouraged to cooperate and collaborate with their peers. This includes seeking assistance from and providing help to peers. Between April and November there was a significant reduction in the amount of help which students in the Above Average group perceive to be receiving from their friends; in the Average group there was no

change and for the Below Average group there was a small increase in the help received. This finding is not surprising, as it might be expected that the computing skills of many of the students in the Above Average group are stronger than those of students in other groups. The Below Average students are likely to feel an increasing need for assistance and obviously receive it.

With respect to providing help to others few of the Below Average students saw themselves as helping their friends in April while 85% of Above Average and 70% of Average students helped their friends. In November this difference is no longer significant and three-quarters of students in both the Average and Below Average groups are helping friends with their computing while 91% of the Above Average group do. Many of the students in the top ability range are regarded as *experts* and expected by their teachers to do a considerable amount of the teaching.

The self-reliance of students in the Above Average group is also reflected in their reactions to specific problems. Irrespective of specific subject contents most of them *keep trying* and when really unable to solve the problem ask someone for help (rarely the teacher) and very few of them *give up*. Approximately 50% of the Average group also keep trying and then *watch someone else working on the task*. Except in the area of computer programming these students ask for help significantly less frequently than those in the Above Average and Below Average groups. Unfortunately, the vast majority of students in the Below Average group just *keep trying* without success. On the whole only 15% of them ask someone for help (in this case mostly the teacher). However, half of the students in this group ask for help (from the teacher) when they encounter difficulties with programming. The picture is different with respect to technological problems. When the printer, disk drives, etc. cause difficulties 66% of Above Average, 62% of Average and 83% of Below Average students seek the teacher's help. There has been no significant change in these pattern, of seeking help between April and November.

With respect to problems encountered in mathematics 40% more of the Above Average and Average students seek help from a friend rather than the teacher in November compared with April. Less than 10% of the Above Average and only 11% of the Average students ask the teacher for assistance. Equal proportions of the students in the Below Average group turn to friends, class *experts* and the teacher. The pattern of asking for help in science is very similar to that in mathematics.

In writing (word processing) and reading students in the Above Average and Below Average groups ask family members for assistance more frequently than the teachers. Students in the Average group ask their teachers in these areas 10% more frequently than family members. Class *experts* are asked less frequently than family members in all groups.

Students' Expectations and Perceptions of Teachers In November 29% (April: 32%) of Above Average, 19% of Average (April: 36%) and 17% (April:

57%) of Below Average students stated that the fact that students have their own laptop computers does *not* mean that they require less attention from the teacher. The widespread fallacy that technology might actually replace teachers or at least reduce their importance in the classroom is not supported by these students. The large decrease in the perceived need for teacher assistance among the students in the Average and particularly Below Average groups is interesting, as many of them also report not to be coping well with computing. At the same time 62% (April: 42%) of Above Average, 42% (April: same) of Average and 46% (April: 36%) stated that computers will never replace teachers. Might these students have given up on their particular teachers?

Half of the students in the Above Average and Average groups and 80% of those in the Below Average group feel that it is difficult for teachers to find the time to teach students. There was a significant increase in this feeling in all groups between April and November. At the same time all groups report that the teachers are spending less time keeping the equipment in order in November than in April. Could it be that the teachers, who are obviously more experienced in November than they were in April, are less sure of what and how to present to the students as the project continues? Lack of pre-service training and staff development opportunities might be catching up with them.

There is a highly significant increase, between April and November, among Above Average and Below Average students in the view that *teachers should know all the answers about computers and computing*. 67% of the Above Average, 81% of the Average and 92% of the students in the Below Average group wish that the teacher could answer more of their questions. One quarter of the students in all groups believe that computers can teach better than teachers. All students believe that computers can help students and teachers to work and learn together, and that students can teach teachers things about computing. However, while all but a few Above Average and Average students (at least sometimes) enjoy teaching the teacher something new about computing, 31% of the Below Average students, who report to have been able to teach their teacher, just do not enjoy it.

Conclusion It is obvious that the students are less excited and perhaps more realistic about their laptops in November than they were in April. A major concern is that the Year 7 students, who have been part of the project for two years, are becoming increasingly despondent about a number of major aspects of their experience in the SUNRISE classroom. There is no decrease in enthusiasm regarding computers and computing *per se* -- rather the students are becoming increasingly more dissatisfied with their personal experiences in educational computing.

Many of the students in all three intelligence categories appear to need and expect more assistance with their computing from their teachers, although the Below Average students might require such assistance most of all.

Computer awareness is high in all groups, but there are indications of higher levels of frustration among students than expected. Students still have considerable enthusiasm for and faith in computers in themselves, but personal application is lower than might have been expected. The fact that significantly fewer students, even in the Above Average IQ group, look for library books on computers and computing, and that fewer students are interested in reading books about computing in November than in April -- even though there is a belief that the teachers are unable to answer sufficient questions relating to computing -- might well be interpreted as a lack of commitment on the part of the students to their computing.

Educational Achievement

Towards the end of Term 3 a small battery of standardised general achievement tests was administered to all students in the sample and to 60 Year 6 and 59 Year 7 students in parallel classes which are not part of the SUNRISE project and thus do not use laptops in their work. The reason for this assessment was to provide an estimate of the relative levels of achievement attained by students in the SUNRISE project with respect to basic learning areas. It was regarded as important to establish whether the Year 6 and Year 7 SUNRISE students happened to be a particularly advanced group, or whether they were actually pretty comparable with students in other schools. A second, perhaps even more important reason, for this assessment was to investigate whether, when required to perform without their computers, children who have permanent access to computers for all their work will show that they have developed traditional literacy, numeracy and writing skills.

The tests used were those that make up the *ACER Australian Cooperative Entry Program (ACEP)* (ACER, 1991), a secure battery prepared by ACER annually and used for a variety of purposes. All tests in this program are administered under standard conditions, and coded and scored at ACER. Norms are available for special populations and more general samples in most Australian states. The tests have not been used in Queensland before. The battery was administered by Irene Brown, assisted by the teachers, under the required standard conditions. All tests but the essays were machine scored at ACER. The essays were scored by two markers from the usual ACEP team, i.e. each essay was assessed by two examiners independently.

The ACEP battery aims to assess performance in six areas as follows:

1 *Reading*. In this test students are required to read short passages on different topics, and answer questions about each passage to show that they understand what they have read. The student chooses between four alternative answers for each question.

Humanities -- Comprehension and Interpretation. Students inspect printed text and pictorial material and answer questions by choosing from four alternatives. The content of the items is drawn from areas such as English, art, history, geography and social studies. No prerequisite factual knowledge is assumed.
Mathematics Comprehension. Students are asked to solve a number of problems using the information given in the question and to apply the basic mathematical skills which they have acquired at school. No complex or difficult calculations are required.

4 *Language Usage*. Spelling, word usage, sentence construction, punctuation and capitalisation are assessed.

5 *Mathematics Achievement*. The content of this test relates to three areas: (a) basic number operations, fractions, decimal fractions and percentages; (b) measurement of the length of line segments and the area of the region enclosed by regular figures; and (c) the interpretation of simple graphs.

6 *Written Expression*. One topic is set for all candidates. Within 25 minutes students are asked to plan their response as well as to write about the topic. Planning space is provided as well as 1½ pages of ruled paper for writing. Students are not asked to make their piece of writing a special length. The directions ask for *clear, lively, vivid and interesting writing.*

In the total group of 233 students (i.e. 115 SUNRISE students and 118 non-SUNRISE students) the mean scores of Year 7 students in all six areas of assessment were higher than those of the Year 6 students, but these differences were not statistically significant. There was no significant difference between the levels of performance of the students in the SUNRISE and non-SUNRISE classes. In the non-SUNRISE group the mean scores of the girls were higher than those of the boys, and often significantly so. An exception to this occurred in the test of *Mathematics Comprehension*, where the boys performed better than the girls.

In the SUNRISE group, Year 7 students obtained higher total scores than Year 6 students, and the mean total score for girls was significantly higher than that of the boys ($p<.05$). Girls performed better than boys within each year in all areas except in the test of *Mathematics Achievement*, where Year 7 boys performed significantly better than any other group. Year 6 girls performed marginally better than Year 7 boys in *language usage, written expression* and *mathematics comprehension* . As noted above, Year 7 boys tended to obtain the highest scores in mathematics achievement; they also obtained marginally higher mean scores in *reading* and in the *humanities*.

The gender differences noted above are found in all samples with whom the ACEP has been used. They reflect general trends in educational achievement in our society, which actually increase with the age of the students.

The mean total performance and the average in each assessed area of the total sample and the subsamples was at the upper end of the average range. Only in the weaker students did performance in computing correlate with achievement test

performance. What is interesting and important is that, on the whole, the SUNRISE students performed no better or worse than the students in the classes not provided with laptops. This suggests that the enrichment provided by learning to work with computers does not disadvantage the students when they are required to work without computers. Other tests, especially longitudinal measures, would have to be used to assess positive generalised achievement outcomes resulting from the availability of laptops.

R1 Versus R5 Students

As was reported previously, peers and teachers ranked the students in each class according to the effectiveness with which they used their laptops. R1 students were judged to be the *best students in computing* by teachers and peers, while the R5 category contained the students whom their teachers judged to be doing badly in computing.

Of the 115 students in the SUNRISE classrooms, 27 (23%) were categorised as R1, 30 (26%) were rated as R5, and 58 (50%) were given a middle rating. As can be seen in Table 7.3, the proportions of students in each rating category did not differ significantly between the Years. Table 7.4 shows the mean IQ for students classified as R1 and R5.

Gender Differences As is shown in Table 7.4 more boys than girls were rated to be doing well (R1) and doing badly (R5). The trend of rating proportionately more males than females as either 1 or 5 was similar within year levels, the exception being the larger proportion of Year 7 girls than boys who were rated as doing badly (R5). The implications of these gender differences are discussed further in Chapter 8.

Differences in Feelings and Attitudes of R1 Versus R5 Students

Anxiety There were significant differences between R1 and R5 students with respect to manifestations of computer anxiety. At the beginning of the school year 50% of the total sample were still *scared that they might break their computer*. Even though Year 7 students had been using computers for 14 months longer than Year 6 students at that time, their anxiety level was higher than might have been expected. At the end of the school year about a quarter of all students were still frightened of their computers. R5 students appeared to be less anxious than their peers. Only 18% of R5 students reported anxiety about their computers in November. This shows that the reasons behind reported anxiety may differ for individual students and groups. It might also suggest that R5 treasure their computers less than R1 students.

Table 7.3 Ratings for Student Achievement in Computing

	Rating		
	R1	R5	R2-R4
Total	27 (23%)	30 (26%)	58 (50%)
Total boys	16 (27%)	18 (30%)	26 (43%)
Total girls	11 (20%)	12 (22%)	32 (58%)
Year 6	13 (23%)	15 (27%)	28 (50%)
Year 7	14 (24%)	15 (25%)	30 (51%)
Year 6 boys	8 (27%)	12 (40%)	10 (33%)
Year 6 girls	5 (19%)	3 (12%)	18 (69%)
Year 7 boys	8 (27%)	6 (20%)	16 (53%)
Year 7 girls	6 (21%)	9 (31%)	14 (48%)

Table 7.4 Mean IQs for R1 and R5 Students

	N	Mean	Standard Deviation
R1 Total sample	27	113.81	13.02
R5 Total sample	30	94.23	8.38
R1 Year 6	13	114.38	13.67
R5 Year 6	15	94.60	8.73
R1 Year 7	14	113.29	12.89
R5 Year 7	15	93.87	8.30
R1 Year 6 girls	5	115.60	19.83
R1 Year 6 boys	8	113.63	9.69
R1 Year 7 girls	6	120.17	11.51
R1 Year 7 boys	8	108.13	11.95
R5 Year 6 girls	3	96.00	10.39
R5 Year 6 boys	12	94.25	8.76
R5 Year 7 girls	9	96.89	6.58
R5 Year 7 boys	6	89.33	9.07

Some students would feel a certain amount of anxiety about being responsible for an expensive piece of equipment. They are taking their parents' and teachers' warnings to treat their laptops with care very seriously. Other students appear to have a fear that the computer will do things their owners had not intended; in other words, these students are anxious because they feel that they are not in control on their computers.

In April 36% of Year 6 and 29% of Year 7 R1 students and 47% of Year 6 and 33% of Year 7 R5 students reported that computers still frighten them. In November the proportion of students still apprehensive about the hardware reduced significantly among Year 6 R1 students and for all R5 students. In November 23% of Year 6 and 29% of Year 7 R1 students and 14% of Year 6 and 21% of Year 7 R5 students reported that the computer still *scares* them.

In April 68% of R1 reported not to be scared at all, while in the rest of the total sample only 37% reported not to be scared at all. In November one quarter of both Year 6 and Year 7 students reported still to be scared of their computer. 26% of R1 and 18% of R5 students reported still to be scared.

Approximately half of all students are still frightened that they might break their computer, 71% of Year 6 R5 as against 54% Year 6 R1, 40% of Year 7 R5 as against 29% of Year 7 R1. These figures probably reflect a reasonable concern on the part of the children for expensive property.

After two months of the 1991 school year 87% of students in Year 6 who were classified as R5 (this reduced to 71% in November), but only half of the Year 6 R1 students (54% in November), reported that personal computers used to *scare* them before they got used to them. In the total sample of Year 6 and 7 students 40% (in November 49%: 37% of R1 and 54% of R5) reported to have been scared initially while 57% of R5 in the total sample reported to have been scared.

Enthusiasm The school reports that approximately 10% of the students chose to leave their computers at the school overnight because they do not wish to use them at night, in some cases because they are frightened that they might be damaged on the way home. In April nearly one-third (this reduced to a quarter in November) of the Year 7 R5 students did not like the fact that they are encouraged to take the computer home at night. In April R1 students took their computers home every night and were very happy with this arrangement. In November 30% of R1 and 25% of R5 students wish that they did not have to take their computer home every night. The majority of R1 students who object to taking their computer home at night are in Year 7.

Among R1 students 77% (November: 70%) in Year 6 and 64% (November: 68%) in Year 7 indicated that they would be *very unhappy in a school without computers*. In April all R5 students indicated that they would be unhappy in a school without computers. In November 79% of Year 6 and 57% of Year 7 R5 students felt they would be unhappy in a school without computers. In the total sample in April more than 80% of students indicated that they would be unhappy in a school without computers. This proportion reduced to 74% in November.

There was no significant difference between R1 and R5 in April in response to the statement *When I am an adult I will get a job where I can use computers*. In November 77% of R1 and 86% of R5 in Year 6 felt they would. But only 31% of R5 in Year 7 did (85% of R1 in Year 7). This is another indication of the alienation of Year 7 students observed throughout the study.

The same is borne out by the fact that in April 31% of R1 (25% of Year 6 and 36% of Year 7) and 45% of R5 (57% of Year 6 and 33% of Year 7) students tried to find books about computing in the library, but in November 58% of R1 and 50% of R5 in Year 6 try to find books, while only 21% of R1 and 7% of R5 do in Year 7.

In April all R1 students in Year 6 and 79% of R1 students in Year 7, and 80% of R5 students in Year 6 and 73% of R5 students in Year 7 found it interesting to figure out what went wrong with their computing. In November 92% of R1 and 79% of R5 in Year 6 found this interesting, while only 64% of R1 and 71% of R5 in Year 7 did.

Self-image and Confidence In both years the confidence of students that they would get their work done better with a computer was high. In April R5 students tended to be even more confident than R1 students; the confidence in R5 and Year 7 R1 students was reduced in November. It is likely that R1 students were more realistic from the beginning, while R5 students might have believed in certain *magic* powers of the computer. In the total sample, close to 70% of students believed in April that the computer would cause them to do their work well. This proportion rose to 77% in November. Despite high enthusiasm, R1 students appear to be more realistic in their assessment of the usefulness of computers than R5 students. This view is supported by the finding that over 70% of R5 students in both Years 6 and 7 felt that children who use computers *often* do better in their schoolwork while R1 students in both years and the majority of the total sample felt this to be true *sometimes*. In November 100% of both R1 and R5 in Year 6 reported that using the computer makes mathematics easier, but only 57% of R1 and 71% of R5 in Year 7 did.

In November 43% of R5 students but only 7% of R1 students feel that they are not the type to do well with computers. In the total sample 18% of Year 6 and 32% of Year 7 expressed the same lack of self-confidence.

Enthusiasm about using computers and the perception of the power of computers thus did not appear to be related to level of achievement in computing, but confidence and self-image with respect to expertise in computer use are. Also, there was definitely a decrease in enthusiasm during the school year in both Years, particularly in the academically weaker students and those who had not kept up with the rest of the class with respect to the acquisition of computing knowledge and skills.

Knowledge of Computers and Computing In April most students felt that computers are as intelligent as people. However, in November two-thirds of both R1 and R5 in Year 6 disagreed with this view, while all R5 students and all but one R1 student in Year 7 think that computers are *sometimes* as intelligent as people.

At least 90% of students in all groups agreed that most people in the community do not understand what computers are about. The majority of both

R1 and R5 students and the total sample regard computers as less intelligent than people, but in Year 7 significantly more R5 students than R1 students regard computers as more intelligent. In November 77% of R1 and 71% of R5 in Year 6 felt that people are more intelligent than computers and all R1 and all but one R5 in Year 7 did. Everyone in Year 7 and but only 20% of R5 in Year 6 disagreed with the statement that computers will control the universe.

More R1 students than R5 believe that working with the computer helps them in their learning, and a larger proportion of R1 students (53% vs 31% for rest 2/3) believe that parents should send their children to schools which have computers. Two-thirds of R1 but only one-third of R5 students (and 46% of the total sample) believe that every school child and adult should be able to use personal computers. Significantly more R1 students than R5 students believe that the invention of computers has improved people's lives.

In April, 60% of R1 and 67% of R5 students believed that *one day computers will replace teachers*, but in November 50% of R1 and 86% of R5 in Year 6 but only 43% of R1 and 36% of R5 in Year 7 believe this.

Learning and Problem Solving In response to the statement *Using personal computers makes schoolwork more difficult*, 33% of R1 and 27% of R5 students answered *never* in April. In November 69% of R1 and 57% of R5 in Year 6, but only 21% of R1 and 26% of R5 in Year 7 said *never*.

In April significantly more R1 than R5 students and approximately 40% of the total sample believed that one can use computers without understanding how they actually work. In November this proportion has increased for Year 6 R1 and R5 to 61% and 57% respectively and for Year 7 to 85% and 57% for R1 and R5 respectively.

In April 85% of R1 and 69% of R5 students felt that using the laptop allows them to work independently. In November 85% of R1 but only 43% of R5 in Year 6 agreed, and in Year 7 64% of R1 but 79% of R5 agreed. When it comes to the crunch, R5 rely on the magic of the computer rather than on their own ability and skill. They certainly feel the computer helps them, and think they would be worse off without it. Whether the reason for this is that they are accepting less critically than R1 students what they have been told by their teachers about the usefulness of the computer, or whether they really experience that they are doing better with computers, i.e. would do even worse without, cannot be established here.

In April among R1 students 23% of Year 6 and none of Year 7 felt that they do not have control over what they do when they are using the computer. In November 15% of Year 6 and 7% of Year 7 R1 students, and 7% of Year 6 and 21% of Year 7 R5 students felt that they do not have control over what they do when they are working with the computer. Only 60% of R5 but over 80% of the rest of the sample feel that they have control over what they do on the computer.

In April more than 80% of R1 students in Year 6, and all R1 and R5 students in Year 7, regard the computer as a tool, just like a hammer or a lathe. Only 47% of the R5 students in Year 6 saw the computer as a tool. In November 93% of R1 and 75% of R5 students regarded the computer as a tool. At the end of the school year all R1 and R5 in Year 6 but only 57% of R1 and 71% of R5 in Year 7 agreed that using computers makes learning mathematics easier.

Learning how to use computers was regarded as hard work by all students. *Figuring out what went wrong in my computing* was regarded as interesting more frequently by R1 students than by the rest. In November only 8% of R1 and 21% of R5 in Year 6 felt that figuring out what went wrong with one's computing is never interesting, but 36% of Year 7 R1 and 29 of Year 7 R5 thought so.

A significantly larger proportion of R1 than R5 students in both Years 6 and 7 believe strongly that children who use computers do better in their schoolwork. Approximately 60% of students in the total sample feel that students having got used to working with computers will find it difficult to work without them. There was no significant difference in this view between R5 and the rest of students, nor between R5 and R1 in April. In November 61% of R1 and 57% of R5 in Year 6 but only 21% of R1 and 43% of R5 in Year 7 agreed with this.

Control 93% of R1 but only 64% of R5 students feel that having a laptop puts them in charge of their own work. In November 15% of Year 6 and 7% of Year 7 R1 students, and 7% of Year 6 and 21% of Year 7 R5 students felt that they do not have control over what they do when they are working with the computer. Only 60% of R5 but over 80% of the rest of the sample feel that they have control over what they do on the computer. On the other hand, only one R1 student but a quarter of the R5 students believe that they are *never* the boss of their computers.

Favourite Activities Using Computers As noted previously, favourite activities named by the students were more frequently programming related rather than discipline related, except among the Year 7 R5 students. When asked to name the three best things about using laptops 70% (April: 90%) of R1 students and 75% (April: 100%) of R5 students named activities relating to learning, knowledge and skills development. Only 4% (April: 22%) of R1 and 25% (April: 20%) of R5 students mentioned aspects of computing which relate to the physical presentation of school work, particularly the fact that typed work is more pleasing than handwritten work. The greater chance for social interaction between students during learning was named by 48% (April: 33%) of R1 and 57% (April: 27%) of R5 students. Twice as many R1 as R5 students named activities reflecting interest and curiosity with respect to the novelty aspects of the technology. 19% of R1 students focused on time-saving and convenience aspects of computer use in both April and November, while 32% (April: 17%) of R5 did. One quarter of the R1 and none of the R5 students named computing as an asset for the future and increased job opportunities, while 41% of R1 (April: 48%) and 25% (April: 32%) of R5 students simply said that computing is fun.

No significant differences were found between R1 and R5 students with respect to the subject areas in which individual students like using computers best. The smallest proportion (i.e. 2 R1 students and 3 R5 students) chose mathematics. No R1 student chose science. Social studies was chosen more frequently by R1 than R5 students. R5 students chose language, writing and social studies in equal proportions.

There were no significant differences between R1 and R5 students with respect to the subjects liked less now that the students are learning with computers. 63% of R1 but only 29% of R5 students report that there are no subjects which they are liking less now that they are working with laptops.

Help R1 and R5 students in both years discuss computing, ask help from friends, parents, etc. and offer help to their friends with similar frequencies. However, significantly more R1 than R5 students in Year 7 seek help from their parents. In November there was no significant difference between R1 and R5 and Year 6 and 7 with respect to the proportion of students discussing computers and computing with parents, other family members and other adults.

When *stuck* in their work, R1 tend to keep trying, and when they really cannot solve the task (e.g. in mathematics) they use paper and pencil first. None of the R5 keep trying nearly half of them in fact abandon the task. One quarter of R5 students ask the teacher for help. R1 students ask help from friends and class *experts* more frequently than from the teacher. A reason for this may be that they have understood that the teachers would like them to work independently and ask peers before they request help from the teacher. Also R1 students are perhaps more readily able to identify the experts in the class.

Students' Expectations and Perceptions of Teachers In April 4% of R1 and 13% of R5 students felt that *Teachers should know all the answers about computers;* in November 39% of R1 and 63% of R5 in Year 6 and 21% of R1 and 7% of R5 in Year 7 felt this way. The changes in student attitudes are likely to be the result of teaching practices in the SUNRISE classrooms as much as of student adaptation to learning with computers. As the year progressed students at all levels of achievement became more frustrated with the amounts of help they were receiving from their teachers. Only a small proportion of R1 students reported that they enjoy teaching their teachers new things about computing. The R5 students did not enjoy this; obviously, they would have less occasion to do so.

In April 70-80% of both R1 and R5 students (compared with 60% in the total sample) believed that one day computers will replace teachers. In November 50% of R1 and 86% of R5 in Year 6, but only 43% of R1 and 36% in Year 7 agreed. Might this reflect some wishful thinking on the part of some of the students?

Significantly fewer R5 than R1 students in both years felt that students who use computers need less attention from the teacher. Significantly more R1 students (approximately 90%) than R5 students (70%) appreciate that personal computers put students in charge of their own work and allow them to work

independently. In November, in the overall sample 74% of R1 and 61% of R5 appreciate this. 85%of R1 and 43% of R5 in Year 6 and 64% of R1 and 79% of R5 in Year 7 believe that using a laptop allows students to work independently. R5 students appreciate the fact that they are able to work independently even more than R1 students. A reason for this may lie in the fact that they are confronted with their mistakes and failures less frequently. While R5 students enjoy the freedom provided by instructional computing, it is likely that they really do need very much teacher guidance and supervision. Are teachers aware of this? It is possible that the teachers at Coombabah have, due to their extra load, disregarded individual differences in the classroom more than they would if they were teaching in a conventional way. The computer may yet underline rather than diminish individual differences.

80% of R5 and 100% in the rest of the total sample accept that teachers do not know everything about computers . 73% of R5 students (80% in Year 6, and 67% in Year 7) expect all teachers to be experts in the use and programming of computers, as against 31% in the total sample (30% Year 6 and 32% in Year 7).

50% to 71% of R1 students (60% in the total sample), but only 15-30% of R5 students perceived it to be difficult for teachers to find the time to teach students because they are too busy looking after technical failures and problems with the hardware. For both R1 and R5 the proportion of students feeling like this was greater for Year 7.

Conclusion Right through the study there is considerable evidence that for Year 7 students first impressions of learning with computers, i.e. impressions gained in the chaotic initial days of 1990, when teachers were less competent and confident with teaching with computers (cf. Ryan, 1991), had a lasting effect. Significantly more R1 and R5 students in Year 7 than Year 6 students felt that keeping the computers, printers, etc. going takes a lot of teachers' time. More R1 than R5 students believe that teachers do not need to know everything about computers and computing. As was noted above, R5 students see themselves as requiring more help from the teacher, and hence expect greater expertise on the part of the teacher. R1 students are generally more confident themselves, and have accepted that the Sunrise classroom provides an opportunity for collaborative learning between students and teachers. R5 students would be expected to be less accepting of the latter idea.

WORKING ON COMPUTER PROGRAMMING TASKS

To gain some indication of the level of students' knowledge and understanding of computer programming and programs, their performance on six tasks was analysed. Five of the tasks, all for individual performance, were given within one week at the beginning of Term 4 of the 1991 school year and the sixth, a group task, one week later. Three of the tasks were program production tasks and three

were essentially program comprehension tasks. The production tasks were aimed to assess programming proficiency, which obviously requires the knowledge and some understanding of programming rules. The tasks were also designed to assess the students' skills in problem decomposition, planning and the use of procedures. In addition, the program comprehension tasks required a deeper understanding of the overall structure of the programs. The tasks were as follows:

Production tasks:	Comprehension tasks:
Task 1: Numbers 1-20	Task 4: Two flags
Task 2: Diamond	Task 5: Robot
Task 3: Rectangular shapes	Task 6: Castle

Sample. Not all children participated in all tasks. Students were absent for a variety of reasons. All students took part in Task 1. Only 105 (51 male and 54 female) students participated in Task 2 and 91 (42 male and 49 female) students participated in Task 3, 107 (53 boys and 54 girls) in Tasks 4 and 5, and 101 (54 boys and 47 girls) in Task 6.

The sample for Task 2 consisted of 52 Year 6 students (27 boys and 25 girls), and 53 Year 7 students (24 boys and 29 girls, i.e all the Year 7 girls). The sample for Task 3 consisted of 41 Year 6 students (17 boys and 24 girls) and 50 Year 7 students (25 boys and 25 girls). Tasks 4 and 5 were attempted by 53 Year 6 students (27 boys and 26 girls) and 54 Year 7 students (26 boys and 28 girls). The sample for Task 6 contained 51 Year 6 students (28 boys and 23 girls) and 50 Year 7 students (26 boys and 24 girls).

PRODUCING PROGRAMS

Tasks 1 and 2: Two Simple Production Tasks

In Task 1 the students were instructed to write a program for the computer to print out the numbers one to twenty. Task 2 required students to write the necessary procedures for the computer to draw a diamond. It was emphasised that these procedures were to be the most elegant or efficient that they could work out. In order to encourage the students to think creatively about the task they were also instructed to write as many different procedures as they could. The students were allowed thirty minutes in which to complete both tasks.

The analysis of the resulting procedures focused on the style of procedures, particularly the use of the *repeat* command, the use of subprocedures and recursion or the *make* command. The use of these commands requires higher order thinking, that is, an awareness of programming efficiency and prior analysis.

Table 7.5 Number of Programs Produced in Tasks 1 and 2

	% Students				
Programs	Total	Year 7	Year 6	Boys	Girls
Task 1					
1 program	33	38	29	33	33
2 programs	33	30	37	33	33
3 programs	16	13	19	16	17
4 programs	9	9	8	10	7
5 programs	8	9	6	8	7
Task 2					
0 programs	5	9	0	2	7
1 program	54	66	42	51	57
2 programs	23	13	33	20	26
3 programs	12	8	17	18	7
4 programs	5	4	6	8	2
5 programs	1	0	2	2	0

Number of programs produced. One of the instructions given to the students was to make as many different programs as they could for each task. Although some students varied the programs simply by writing the command *show* instead of *print*, about half of the sample wrote different kinds of programs. Table 7.5 shows the numbers of programs produced by the students.

For Task 1, producing numbers 1-20, one-third of the students wrote only one program, and another third two programs, while for the diamond over half wrote just one. One-sixth of the students wrote four or five programs for Task 1 and about the same proportion wrote three or more programs for Task 2. While the percentages of boys and girls who wrote various numbers of *1-20* programs were similar, nearly one-third of the boys but only 10% of the girls wrote three or more procedures to draw a diamond. Similarly, comparable percentages of Year 6 and Year 7 students wrote various numbers of *1-20* programs. Only a quarter of the Year 7 students created more than one diamond procedure, but over half of the Year 6 students wrote more than one procedure for this task.

Analysis of programs for Task 1 (1-20). Table 7.6 shows the percentages of students who wrote various kinds of programs for Task 1. About three-fifths of the students wrote a simple line of numbers or a vertical list. Of those who wrote only one program three-quarters wrote one of these straightforward programs. A small number of the Year 7 and about a quarter of the Year 6 students used addition in their programming. More boys used this type of procedure than girls. Although for a few students this may have been merely a way of writing the program differently, the performance of most appeared to be an approximation of a recursive procedure.

Table 7.6 Analysis of Programs for Task 1

	% Students				
Type of program	Total	Y7	Y6	Boys	Girls
Line of numbers	65	72	58	61	69
List of numbers	59	81	37	53	65
Addition used	16	9	23	22	11
Subprocedure used	24	17	31	25	22
Make used	8	13	2	10	6
Recursion used	18	21	15	31	6
Recursion attempt but unworkable	11	2	21	12	11

A quarter of all students, again more Year 6 students than Year 7 students, used a subprocedure somewhere in the program. A small number of students, mostly from Year 7, used the *make* command, a procedure often used by the students instead of the more complex recursion.

The Year 6 students had been learning how to write a recursive counting program in which they set three variables: a beginning, an end and an interval. Consequently, more than one-third of these students recognised the applicability of this style of program and attempted recursion. Thus 15% of the Year 6 students and 21% of Year 7 produced a workable recursive program. This represents almost one-third of the boys but only 6% of the girls. In fact, no Year 6 girl managed to make a recursive program work, although 12% attempted it.

Analysis of programs for Task 2 (Diamond). Although 80% of the students created a simple step-by-step procedure for the computer's drawing of a diamond, only about half of them considered where the diamond was to be drawn and set a starting position. Almost one-third of the students used the *repeat* procedure, thereby lessening the number of steps required. 55% of Year 7 and 37% of Year 6 students wrote a procedure using variables so that the diamond could be drawn in different sizes. Nearly one-fifth of the Year 7 students added an extra refinement (such as messages to the user or a recursive procedure to fill the diamond), while only 6% of the Year 6 students did so. The boys tended to produce more programs than the girls, but there was little difference between the programming efficiency of boys and girls. Table 7.7 shows the different kinds of programs written for Task 2.

Task 3: Production of a Series of Rectangular Shapes

In this task, taken from Kurland, Clement, Mawby and Pea (1987), the students were presented with seven geometric shapes (five of them using the rectangle as a basic form). The students were instructed to write procedures to produce five of

Table 7.7 Analysis of Programs for Task 2

	% Students				
	Total	Y7	Y6	Boys	Girls
Step-by-step procedure	79	70	90	80	78
Set position procedure	51	43	60	51	52
Use of 'repeat'	30	32	27	29	30
Variable used	46	55	37	41	50
Extra refinement	11	17	6	12	11

the seven figures. It was emphasised that these procedures were to be their best -- that is, the most elegant or efficient that they could produce. In order to encourage the students to look at the task as a whole, and plan, they were instructed to write down which figures they would produce and in which order, before they started. They were free to change their choice once they had begun. The analysis of the resulting programs focused on the style of procedures, particularly the use of the *repeat* command, the use of subprocedures and recursion and the *make* command. Time allowed for this task was 30 minutes.

The figures. Figure 7.1 shows the shapes the students could choose from. Five of the seven figures were based on the rectangle shape so that one basic procedure could be used for all of them. The other two were designed so that a step-by-step approach might be seen as the most obvious for their production.

Shapes A and D were designed so that they could not be broken down in other words the *repeat* command could not be used, nor could subprocedures be written with ease. For students of a lower level of programming proficiency these figures are no more difficult than the more symmetrical figures. This was reflected in the choices made by the Year 6 and Year 7 students. The less experienced group, Year 6, chose A and D almost as often as C. However, the more experienced group clearly differentiated between the figures. Whereas figure A was produced by 63% of the Year 6 students, it was produced by 52% of the Year 7 students. Similar results were shown for figure D. Only 5% of the Year 6 students avoided both A and D, whereas 38% of the Year 7 students did so. Table 7.8 shows the percentage of students choosing each figure. Figure 7.2 shows the performance of Coombabah students on Task 3.

Table 7.8 Percentage of Students Choosing Each Figure in Task 3

Figure		A	B	C	D	E	F	G
Year 6	(n=41)	63	53	65	58	75	78	28
Year 7	(n=50)	52	78	80	50	98	90	48
Total		57	66	73	53	88	84	39

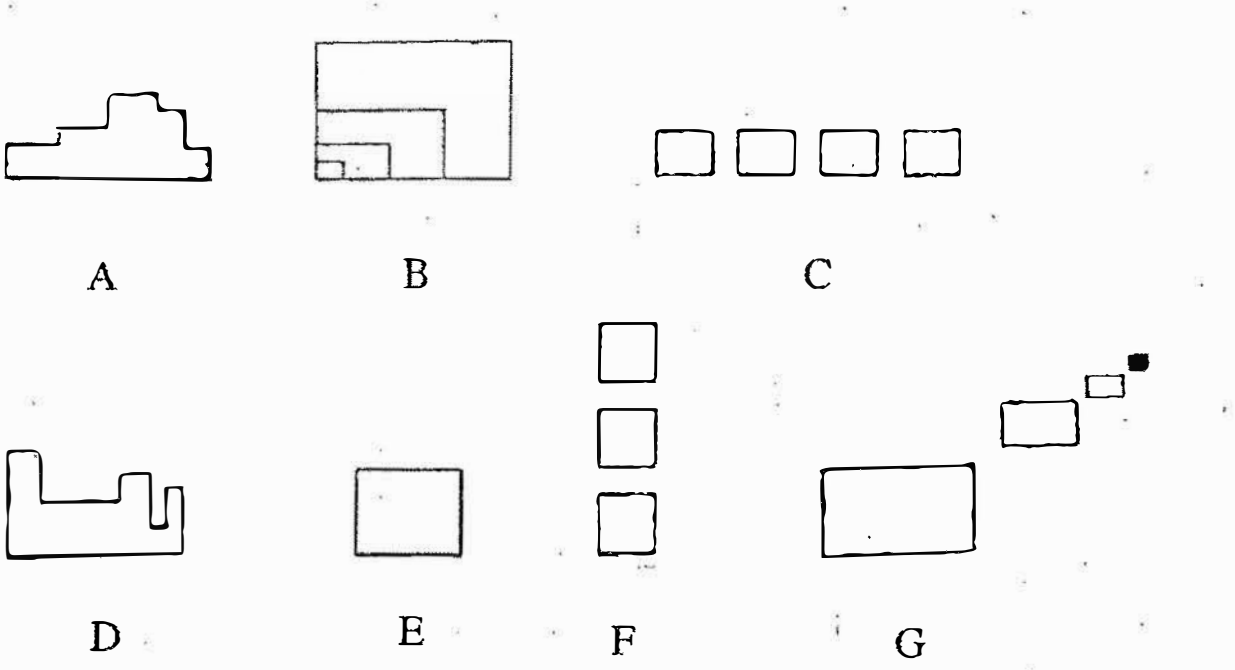


Figure 7.1 The Series of Figures for Task 3

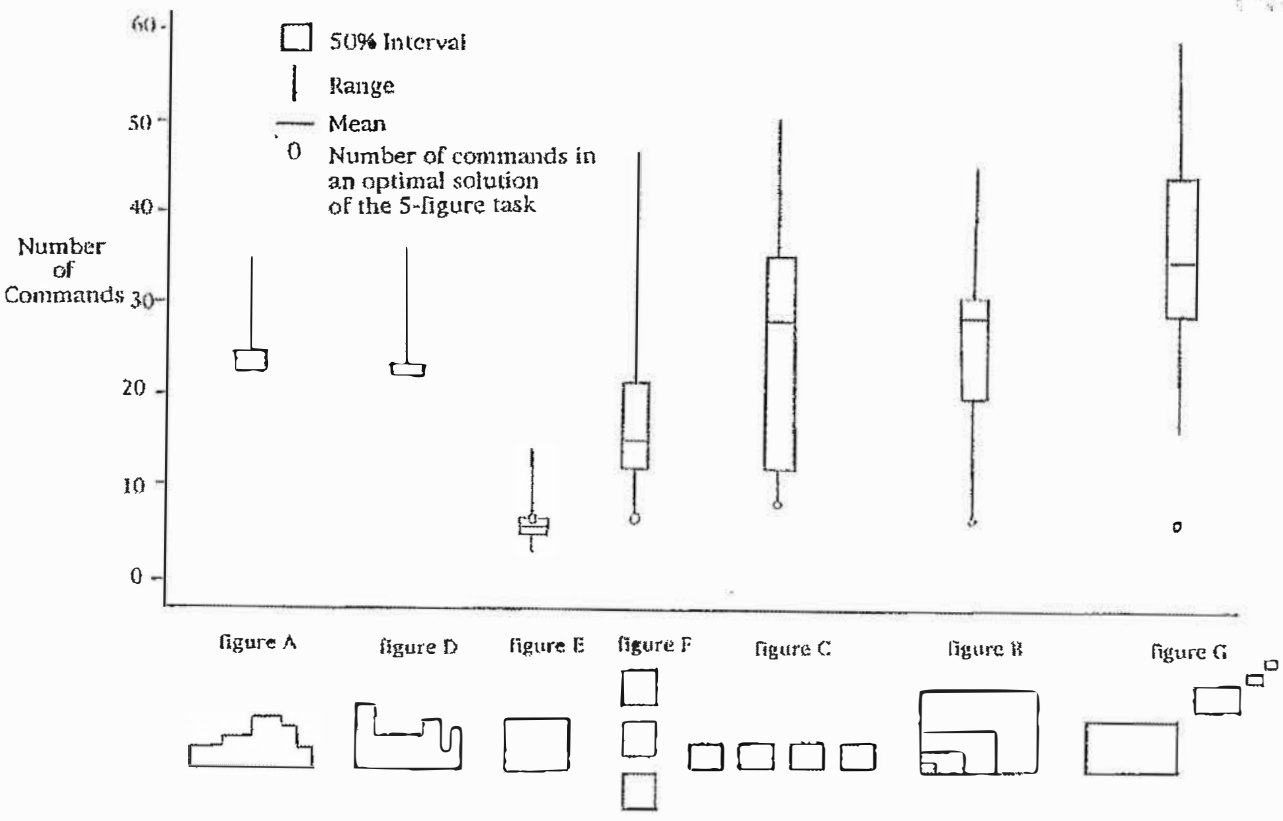


Figure 7.2 Performance of Coombabah Students on Task 3

Choices of the other figures also reflected individual students' understanding. Figure E was the most straightforward, a simple rectangle. This was chosen more often than any other figure, and was done by all but one (one of the top programmers) of the more experienced group. Figure F, three squares, was the next most popular figure, again with almost all the more experienced group choosing it. Figure C, four rectangles in a line, was chosen by 73% of the students, 80% of the more experienced group. This group of figures, C E F, was chosen by 72% of the more experienced group, but only 40% of the younger group.

Finally, shapes B and G both contained a series of rectangles. B was seen to be quite straightforward (four different sized rectangles overlapping) and was chosen by 66% of the students, 53% of Year 6 and 78% of the more experienced group. However, figure G with different sized rectangles placed diagonally was clearly recognised as more difficult, being chosen by 39% of the students, only 28% of the less experienced group attempting it.

Accuracy. The students were supplied with the figures drawn on a grid so that it was possible for them to see the exact proportions of the figures. For figures A and D many students used the grid and produced perfectly accurate copies (53% for A, 50% for D). However, for the other figures the students tended to approximate shapes, producing a rectangle of any proportion rather than of the proportion represented on the diagram, or using the *set position* command for each separate part of a figure rather than using the *forward* command to represent the exact number. Accuracy for these figures was much lower (24-32%), especially for figure G (14%).

Workability. The procedures were analysed for workability. If a figure produced was accurate apart from proportions (i.e. width to length of rectangles, spaces between figures to the size of figures, etc.), the procedure was considered to be workable. If the figure produced was incomplete, at a different angle, or did not work, it was categorised as not workable. The students did produce a high percentage of workable programs -- over 90% for all but figures B (85%) and G (74%).

Programming proficiency. Figures B, C, E, F and G were designed to allow students to use commands such as *repeat*, or to write reusable subprocedures. While many students used linear step-by-step programming for several figures, over 80% used *repeat* at some point. All students used the step-by-step approach for figures A and D. Two students used variables, a higher level procedure.

Figure F, the only figure with squares, was recognised, more than any other, as one for which a *repeat* command could be used. Over 70% used this command for F, and over half for C. The higher level procedures, such as the use of variables, subprocedures and recursion, were used far more sparingly. Of the less experienced group five students wrote a subprocedure for E, and only one used subprocedures for other figures. In the more experienced group, approximately 25% of the students used variables for each of the figures B, C, E, F and G, and 25% used subprocedures on one or more of these figures. Only five students (all

Table 7.9 Percentage of Workable Programs by Shape and Method

Figure	B	C	E	F	G
Step by step					
Year 6	41	67	63	11	67
Year 7	53	24	41	18	25
Repeat					
Year 6	53	29	33	71	33
Year 7	24	32	30	42	40
Variables					
Year 6	-	-	-	-	-
Year 7	12	27	24	17	35
Subprocedure					
Year 6	6	4	4	18	-
Year 7	15	22	4	24	25
Make command					
Year 6	-	-	-	-	-
Year 7	12	-	-	-	20

in Year 7) used the same subprocedure for three or more figures, and four of these used the *make* command, but only on figures B and G. Although no-one used recursive procedures, these students used the *make* command in a conceptually similar way -- that is, they wrote one procedure for the series of rectangles, then used *make* to change the variables.

Table 7.9 shows the method used to make the figure. The percentages reflect the proportion of students who made workable programs. Of those who used the *make* command or used subprocedures, most also used variables (hence the total is over 100%).

Comparison of Performance on Tasks 1 and 2 with Task 3

The 22 students who were regarded as the most efficient programmers (the top 20%) in Task 3 (production of rectangular shapes) also showed efficiency in both Task 1 (1-20) and Task 2 (diamond). One student wrote only simple procedures for Task 1 but wrote a more complex procedure for Task 2, while two students wrote step-by-step procedures for the diamond, but both of these used more complex procedures on Task 1. Had each task had been given separately more consistent programming might have resulted.

Of the 20% of students who scored lowest on Task 3 most also produced fairly simple procedures on the other tasks. However on Task 1 five of this group

attempted recursive procedures and two (both Year 7 students) made these workable.

There was a great variety of results on Tasks 1 and 2 amongst the students who were in neither top nor bottom group in the rectangle series task. In other words, although the top group are producing consistently efficient programs, and the bottom group usually produce unsophisticated programs, the other students are quite inconsistent in their programming efficiency. It appears from these results that students are not generalising the more efficient strategies learnt in one task to other tasks. This was shown particularly by the Year 6 students. Over a third of this level attempted recursion for the Numbers 1-20 task, a similar number used variables for the diamond, one student used the *make* command, and another recursion for the diamond. However no Year 6 student attempted to use recursion, the *make* command, or variables in the rectangle series task. Conversely almost half of the students used *repeat* for the rectangles task but only a quarter used it for the diamond.

These results show that while some students are aware of the more efficient ways of programming they do not apply these consistently. The recursive program in particular was probably recognised as usable by many of the Year 6 students because they had been learning this type of procedure for a very similar task. Yet these students have not yet learnt to recognise other tasks where this procedure is appropriate.

The perception of programming as no more than a tool to achieve certain goals appears to be effective in accomplishing classroom projects. However, the lack of focus on efficiency has meant that most students are inconsistent in their use of efficient programming.

Comparison with the Findings of Kurland et al. (1987)

As noted above, the figures used for these production tasks were described by Kurland et al. (1987). These authors worked with students of Grades 8 to 11 in a six-week summer program. The emphasis of this course was on 'learning to program . . . The teachers . . . tried to bring students to an adequate level of programming proficiency' (Kurland et al., 1987, p. 340). A comparison of the results of the two studies suggests that the emphasis on proficient programming may have been far stronger in the Kurland et al. (1987) project than in the classes at Coombabah.

In their choice of figures the students in the Kurland study revealed an awareness of the difficulty of the procedures involved. Although the more experienced Coombabah group reacted in a similar manner, the Year 6 students appeared to be much less aware of these complexities. Overall the Coombabah students created more workable programs, but much lower levels of programming proficiency than the students in the Kurland study. Kurland et al. do not describe the criteria they used to determine workability, therefore any

comparison is questionable. However, workability as defined earlier in this chapter, showed lower than 90% for figures B (85%) and G (74%) only. These percentages are much higher than those reported for the Kurland sample (47% and 48%) although workability for all the other figures was 80% and above. The short duration of the Kurland program compared with the Coombabah one (six weeks versus eighteen months) might well account for the observed performance differences.

Figure 7.3 shows that for figures A, D and E, which allow little choice for efficiency, the results were much the same between the two student samples. Figures C and F, however, showed different results. The Coombabah students showed a greater range in number of steps used and a higher mean. The 50% interval (which indicates the results of the middle 50% of students) was markedly different between the two samples -- the Coombabah students in this interval generally used more steps. The results from the two most complex figures, B and G, were different again. The range of steps used was similar, but in both cases 75% of the Coombabah students used as many or more steps in their procedures than the bottom 25% of the Kurland students. The Coombabah students seemed to concentrate far more on making workable programs than on programming efficiency, despite the emphasis of the initial instructions. Very few of them (only five Year 7 students) wrote a general rectangle procedure which they then used in three or four other programs. Further, although 80% used the *repeat* procedure somewhere in their programming, only 36% (42% of Year 7, 25% of Year 6) used *repeat* in all of the tasks from figures B, C, E, F and G that they attempted. Clearly these students are not generalising the more efficient strategies used in one task to the others.

These results, although showing excellent programming in a few, suggest that a large percentage of the Coombabah students are not focusing on efficiency, and are not using the higher level thinking skills that Logo can sustain.

Like many other skills, efficient and effective programming might well best be learnt through modelling. One of the disadvantages of the peer teaching approach as practised in the *peer scheme* of the Coombabah project is that the level of expertise and the programming experience of the class *experts* may be just not high enough. These *experts* might be able to produce workable procedures, but they themselves, lacking models and masters (in an apprenticeship sense), are neither efficient nor elegant in their programming. For that matter, as noted before, programming efficiency and elegance are not especially encouraged in the Coombabah classrooms. The important criterion remains *does the program work?*

As the teachers in the Coombabah project were themselves relative novices to programming, programming efficiency and elegance appears to be a relatively low priority for them. As discussed further in Chapter 2, this orientation on the part of the teachers has implications not only for the development of *good programming habits* in students but also for the development (and lack of development) of higher order thinking and problem solving skills.

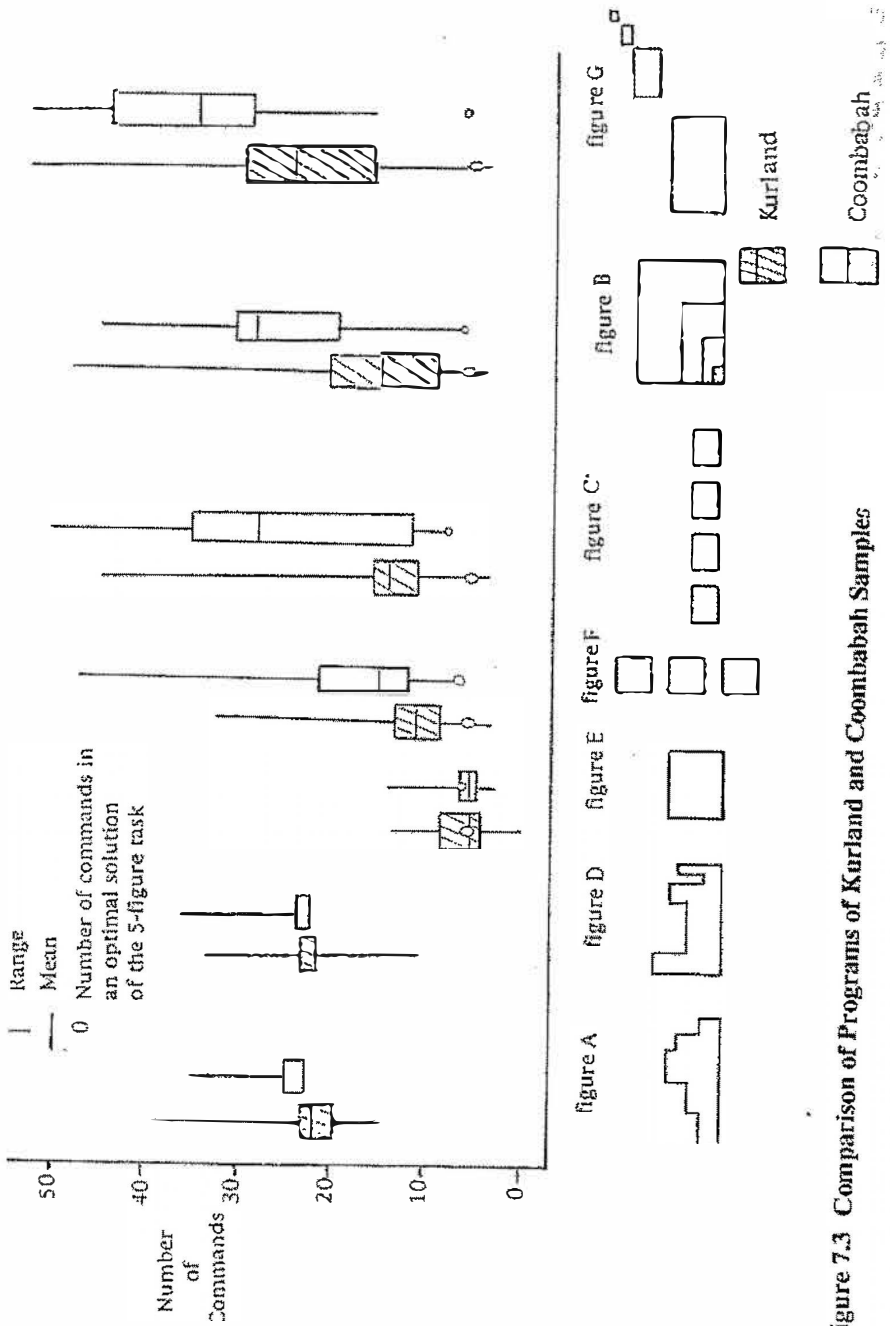


Figure 7.3 Comparison of Programs of Kurland and Coombabah Samples

UNDERSTANDING PROGRAMS

Tasks 4 and 5 were used to investigate students' understanding of computer programming. These tasks, like Task 3, were taken from Kurland et al. (1987) and thus also allowed some comparison of the findings in this empirical study with published research. Each of Tasks 4 and 5 presented four procedures, i.e. one superprocedure and three subprocedures. The students were asked first to write functional descriptions of each of the procedures, and in this way show their ability to comprehend the meaning of commands within the context of a programming procedure. Then the students were asked to draw on graph paper the screen effects of the superprocedure when executed with a specific input. To draw the screen effects, students had to simulate the program's execution. This provided a strong test of their ability to follow the precise sequence of the instructions dictated by the program.

The Kurland procedures were not written in a style commonly used by the Coombabah students. Although the latter write superprocedures referring to subprocedures, they rarely move variables from one to the other, and do not call the same variable by a different name in the subprocedure, as the Kurland program did. Although many of the Coombabah students understand recursion they do not often use it, preferring to use the less elegant *make* command to change the value of variables.

The original programming used by Kurland et al. (1987) was altered somewhat in order to make it a little more understandable to the students in this study. Variables were labelled *length* and *width*, rather than *x* and *y*. The latter would be more familiar to the students in Years 8 and 11 who participated in the original study, but rather more difficult for our Year 6 and 7 students.

The superprocedure was written first, followed by the subprocedures rather than the opposite order as used in the original study. Thirdly, the procedure named *Top* by Kurland et al. was renamed *Head*, as *top* is the name of a primitive in LogoWriter.

Task 4: Program to Draw Two Flags

Each student was given a program sheet with the procedures (see Figure 7.4) and a piece of graph paper with a starting point marked on it. Fifteen minutes were allowed for this task. Although many students had questions, particularly concerning the variables, they were encouraged to work out the procedures by themselves. Table 7.10 shows the results of the comprehension task which involved drawing the two flags.

Almost all our Year 6 and 7 students began correctly with the first flagpole. However, half the students then placed the flag wrongly and/or drew it in the wrong size. Nearly half then drew the second pole correctly, but only 7% were able to draw the second flag correctly. One-sixth of the students were confused

```
Draw Twoflags 10

TO TWO FLAGS :LENGTH
     CENTER
     FLAG 15
     PU RT 90 FD 20 LT 90 PD
     FLAG :LENGTH
END

TO CENTER
     PU HOME PD
END

TO FLAG :LENGTH
     FD 15 BOX :LENGTH CENTER
END

TO BOX :SIDE
     REPEAT 4 [FD :SIDE RT 90]
END
```

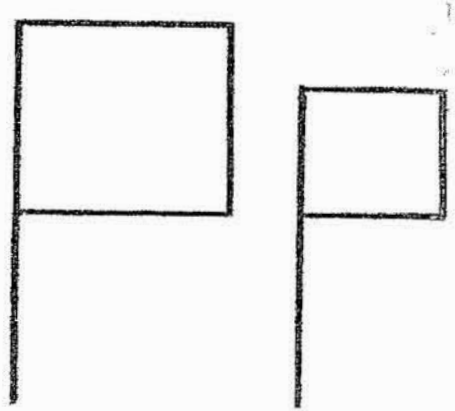


Figure 7.4 Program for Two Flags

Table 7.10 Students' Understanding of Parts of the Twoflags Program

Procedures understood	Year 6 %	Year 7 %	Total %
Flagpole 1	85	92	89
Flag 1	36	56	46
Flagpole 2	38	44	41
Flag 2	7	15	11
Both flags same	16	16	16
Both flags correct	0	6	3

by the change in the variables and simply drew both flags in the same size, most using the variable already within the procedure rather than the one provided in the instructions. Three students, all Year 7, drew the two flags accurately. More or less equal numbers of boys and girls succeeded in the tasks listed. The student performances shown here are roughly cumulative. That is, most students completing a section correctly had also completed the previous one correctly.


```
Draw Robot 6 5

TO ROBOT :LENGTH :WIDTH
     RG HT
     MID :LENGTH :WIDTH
     BK 3 LT 90
     BOT :LENGTH - 2 :WIDTH - 3
     RT 90 PU FD 10 PD
     HEAD :WIDTH - 2
END

TO BOT :LENGTH :WIDTH
     FD :LENGTH RT 90 FD :WIDTH
END

TO MID :LENGTH :WIDTH
     BOT :LENGTH :WIDTH
     RT 90 BOT :LENGTH :WIDTH
END

TO HEAD :LENGTH
     IF :LENGTH < 1 [RT 90 BK 2 STOP]
     REPEAT 4 [FD :LENGTH RT 90]
     FD 1 LT 90
     HEAD :LENGTH - 2
END
```

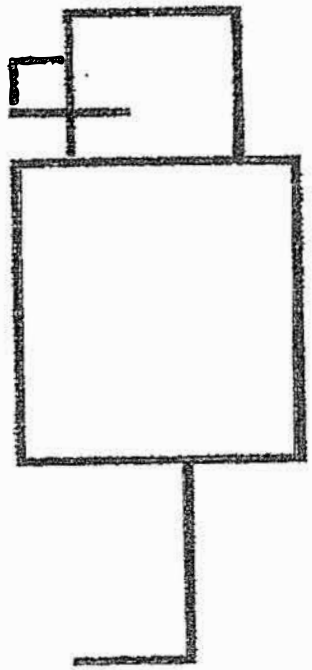


Figure 7.5 Program and Correct Drawing for Robot

Task 5: Robot

The second comprehension task (see Figure 7.5) was a more complex program resulting in a robot. It included moving back and forth between two of the subprocedures, subtracting from the variables, and using a recursive procedure. Time allowed for this task was 15 minutes.

Table 7.11 shows the results of the Robot program. A much higher percentage of the students were able to correctly draw the first rectangle of the program than the other parts. Again the students had difficulty with the variables, making more mistakes with size than with placement, except with the leg, which was misplaced by 27% of the students. This shows that their understanding of the flow of control is reasonable but the use of unaccustomed procedures confused many of them. Nevertheless, a quarter of the students worked out two parts of the body correctly.

Table 7.11 Students' Understanding of Parts of the Robot Program

	Year 6 %	Year 7 %	Total %
Body	56	68	62
Leg	18	20	19
Head	9	18	14
Nose	4	15	9
Mouth	2	4	3
Body, leg, head & nose	0	7	4
Whole robot	0	0	0

Although the performances were again roughly cumulative there were several students at each point who had made a mistake in one part but then produced the next section correctly.

Comparison with Kurland et al. (1987)

Kurland et al. (1987) assessed 79 8 and 11 Grade students. They reported that one of the emphases of their course was efficiency of programming. It might be expected that their students were familiar with the style of programming used in the comprehension tasks. The Kurland sample would thus not have experienced a high practice effect between the two tasks, i.e. Tasks 4 and 5. Table 7.12 compares the performance of these two samples.

While the Coombabah students may have found a number of aspects of the programming style confusing and thus performed less well on Task 4 than the students in the Kurland sample, their performance on Task 5 was much closer to that of the Kurland sample. This may be the result of a significant practice effect for the Coombabah students in Task 5.

While Kurland et al. reported that students reproduced sizes correctly more often than placing shapes correctly, the reverse was true of the Coombabah students. Kurland et al. (1987) suggested: 'Performance on the comprehension tasks showed that students had a fair understanding of individual lines of Logo code but had difficulty in following program flow of control.' (p. 352).

In contrast to this, Coombabah students followed the program through more accurately but had difficulty with the variables, and therefore size, especially in the recursive procedure. These results suggest that many of the Coombabah students are understanding the overall way in which the subprocedures fit together, but were confused by specific aspects of the program.

A further difference between the American and Australian samples was the cumulative effect shown in the performance of the students. Kurland reported less than 3 per cent difference between the cumulative and absolute percentages,

Table 7.12 Comparison of Results for Tasks 4 and 5

Procedures understood	Kurland sample %	Coombabah sample %
TWOFLAGS		
Flag 1 (incl. pole)	48	28
Flag 2 (incl. pole)	21	7
Both flags	19	3
Two flags the same	-	16
ROBOT		
Body	65	64
Leg	37	21
Head	16	14
Nose	13	9
Mouth	2	3

whereas the Coombabah sample showed a difference as high as 9 per cent. These results suggest higher overall understanding of the Australian students, with errors accounting for lower success in earlier sections of the tasks.

Task 6: Castle (group task)

A further program comprehension task was given to the students a week later. This time the students were asked to form small groups. In the SUNRISE classroom most projects and a large proportion of classroom work are carried out in small groups. The children choose their own partners and usually form groups of two to four students. Some students prefer to work by themselves, although none did so for this task. When working in groups the children share their resources and knowledge. Thus, one student may contribute most of the programming or two students may solve problems together. They commonly ask each other, both in their own group and in others, how to program certain procedures or how to debug a program.

For Task 6 students were told to discuss the task in their own group only. The three top-performing students from the other tasks were withdrawn, as it was concluded that they would understand the task correctly and the results of their groups would simply reflect their *expert* knowledge.

The program (see Figure 7.6) was similar to the one for the robot. It contained a superprocedure, named CASTLE, with four subprocedures. Variables needing calculations (e.g. :length / 8) were used and were referred to by different names in the subprocedures. A recursive subprocedure was included. Time allowed for this task was 15 minutes.

```
Draw Castle 20 24
TO CASTLE :HEIGHT :LENGTH
    RG
    BASE :HEIGHT :LENGTH
    FORT :LENGTH / 8
    HOME
    BK :LENGTH / 2
    WINDOW :LENGTH / 6
END

TO BASE :HEIGHT :LENGTH
    BK :HEIGHT RT 90 FD :LENGTH LT 90 FD :HEIGHT
END

TO FORT :SIDE
    REPEAT 3 [LT 90 FD :SIDE LT 90 FD :SIDE
              RT 90 FD :SIDE RT 90 FD :SIDE]
END

TO WINDOW :SIZE
    IF :SIZE < 2 [FLAG STOP]
    PU RT 90 FD :SIZE * 2 LT 90 PD REPEAT 4 [FD :SIZE RT90]
    WINDOW :SIZE - 1
END

TO FLAG
    PU HOME PD RT 90 FD 3 LT 90 FD 3 REPEAT 4 [FD 3 LT90]
END
```

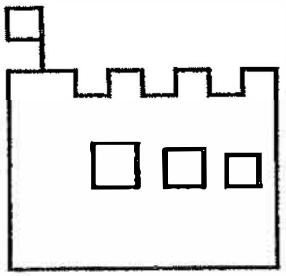


Figure 7.6 The Castle Program

Table 7.13 shows the results of the students' drawings when working in groups. The castle program contained two *home* commands. Students clearly used these to reorient themselves and carry out the following section correctly. The students' performance of this program thus showed much less cumulative effect than the performances on Tasks 4 and 5. For example, 12% more students correctly drew the first window than had drawn the preceding two sections, and 27% more drew the flag correctly than had completed the previous sections correctly. In this program, then, students demonstrated understanding of the separate sections more than they had in previous tasks. In Task 6 more than 75% of the errors were misplacements. In contrast to the errors in the other two tasks, very few mistakes in this task were errors of size. The children appear to have understood the previously unaccustomed use of variables, and this time correctly followed the variable code.

Those who drew the castle correctly were members of two groups, each containing a fairly competent Year 7 programmer. Another two groups correctly

Table 7.13 Students' Understanding of the Parts of the Castle Program

Procedures understood	Year 6 %	Year 7 %	Total %
Base	96	90	93
Fort	33	32	32
Window 1	38	46	41
Window 2	25	11	17
Window 3	25	11	17
Flag	35	42	39
Whole Castle	10	11	9

Table 7.14 Comparison of Students' Understanding of Tasks 4, 5 and 6

Flags	%	Robot	%	Castle	%
Pole 1	89	Body	62	Base	93
Flag 1	46	Leg	19	Fort	32
Pole 2	41	Head	14	Window 1	41
Flag 2	7	Nose	9	Window 2	17
		Mouth	3	Window 3	17
				Flag	39
Flag 1 & pole 1	28	Body & leg	20	Base & fort	29
Flag 1 & pole 2	21	Body, leg, head	6	Base, fort & windows	9
Total	3	Total	0	Total	9

worked out the recursive procedure for the three windows. Interestingly, both the latter groups were composed of four Year 6 students (one group was girls, the other boys). None of these students had correctly worked out either Task 4 or Task 5. One group of five Year 7 students simply drew a castle without any of the required components. All but one of these students had successfully completed at least part of Tasks 4 and 5.

Table 7.14 compares the results obtained on the program comprehension tasks, i.e. Tasks 4, 5 and 6. The results of Task 6, the castle program, show a much higher level of understanding through the separate sections than the other tasks. The number of students who completed the whole drawing correctly has increased, even though the three students achieving best in programming did not take part in this task. However, the cumulative effects are less clear.

Table 7.15 Comparison of Performance on Production and Comprehension Tasks

Procedure	Production %	Comprehension %
Repeat	80	93
Variables	35	94
Subprocedures	25	96
Attempt Recursion	18	41
Correct Recursion	7	17

Comparison of the Results Obtained on the Comprehension and the Production Tasks

In general, students who performed well on the production tasks also performed well on the comprehension tasks. Not surprisingly, the students understood procedures which they were not accustomed to using themselves. Table 7.15 compares the procedures used and components understood by the students on the two types of tasks.

As shown above, recursion was attempted by 18% of the students and completed correctly by 7% in the production tasks. In the comprehension tasks 41% of students understood the first part, i.e. Window 1, correctly, and 17% drew all the windows correctly.

It is evident from the results reported in Table 7.15 that more of the students who accomplished the comprehension tasks used sophisticated programming procedures than those who had been successful in the production tasks. The comprehension of program code requires an understanding on the part of students and makes them think, which in turn leads to more efficient and elegant programming.

Case Study of One Group

The interaction between students in one group was audiotaped as they worked on Task 6, i.e. the Castle program. One member of the group was a class *expert* but he had not performed very well in the other five tasks. The other two boys were not very proficient programmers. The *expert*, Student A, drew the castle while the other two boys, Student B and Student C, took turns in reading out the instructions. Student A occasionally checked the provided procedures. The following conversation took place between the three students in the group:

Student C: *We'll do a bit each, OK?*

They all discussed the program code which they had been given, sharing ideas and correcting each other.

Student C: *Repeat 3 -- left 90 forward 3 left 90.*
Student A: *No, you're doing it wrong -- see look -- you've got repeat 3 left 90 forward three, left 90 again OK?*
Student B: *No left 90's down this way eh?*
Student A: *Yeah -- it's going down because you were up before.*

One of them, Student C, had difficulty with the left and right directions in the Fort procedures.

Student A: *And then does it says left 90 and then go backwards? -- left 90, hang on, we went left 90 here forward 3 and then left 90 again and then we go left 90 again.*
Student C: *I'm getting mixed up.*
Student A: *That's how it goes!*
Student B: *You want me to do it?*
Student C: *I can't understand this.*
Student A: *Hold on, left 90 forward 3 left 90 forward 3. Right 90.*

After the session Students A and B commented: *At the beginning C didn't understand which way the turtle was facing and kept getting mixed up. He understands now.* The students corrected each other and felt free to ask for help where they needed to.

Student C: *Finished?*
Student A: *No way, you've got to help me, guys.*
Student C: *Yeah -- do you just join it up to the thing? How many times have you done it now -- three or four?*
Student A: *One . . . two . . . three*
Student B: *Three times, that's all you're meant to. You're meant to repeat . . . three.*
Student A: *I know, but it's not finished.*

Two of the boys in particular (Students A and C) commented throughout the process on what they thought the castle would look like.

Student C: *I reckon it's only going to make a box.*
Student A: *Oh, I know what this is drawing . . .*
Student C: *The lumpy parts.*

Student A: *The parts that go like that -- dink dink dink dink dink with a flag at the top. I think the flag will go there* [while still drawing the fort].
Student C: *Oh, it's a little square flag.*

One mistake made by several of the more competent groups was to lift the pen at the *home* command. The above described group almost did the same:

Student C: *So you go back to there. It didn't say pen up, did it, so you just go straight -- just go straight -- it did'nt say pen up!*

Another common mistake made by other groups was to start the pattern of the Fort procedure and use it all the way along the top of the castle without checking where it should finish. This group was in danger of making the same mistake.

Student A: *Oh, I know what this is drawing . . . the parts that go like that -- dink dink dink dink dink with a flag at the top.*
Student B: *Where are we up to?*
Student C: *Just keep on doing it because you know what it is.*

These conversations illustrate how individual students would make mistakes which are corrected by members of their group.

The only uncorrected error in this group's drawing was the space between the windows. The boys realised that they had made a mistake when the third window did not fit in, but they did not correct it accurately. The *expert* found a solution and the others allowed him to change it without real explanation. The students reported later: *We only had problems with the windows because we forgot to reduce the distance between them.*

Student A: *The two is right 90 forward 8 -- am I right?*
Student B: *Yeah.*
Student A: *It won't fit.*
Student C: *Ten centimetres square.*
Student A: *If you go size . . . I think this might have been over a bit more.*
Student B: *Doesn't it mean stop all at . . .*
Student A: *Oh, I get it, you told me the wrong thing. It's meant to be in one.*
Student C: *What do you mean in one?*
Student A: *Oh, don't worry.*

One of the aspects of Logowriter which has made it attractive to use with school children is its interactive nature. The students can try a section of programming and find out whether they have done it correctly. However, a consequence of this could be a tendency to approximate and use trial and error.

Student A: *Is it meant to be smaller?*
Student B: *Yeah, it says window minus one.*
Student C: *We should try this on our computer.*

The tendency to guess rather than follow through accurately step by step was shown in the conversation about spaces between windows, as well as by the suggestion to *just keep on doing it because you know what it is*. This attitude was also reflected in the performance on Task 3 (rectangular shapes programming), where a high proportion of the students wrote workable programs, but few wrote accurate ones. As the focus of the project is on learning while using the computer as a tool, rather than on learning to program, students treat programming functionally rather than as an end in itself.[1]

This functional approach is illustrated further in the day-to-day programs children are using. Many of their projects include quite complex procedures. Some students would understand these and could formulate them or change them, but many others simply know them well enough to make use of them. Working in groups enables the students to make use of each other's knowledge without every student having to understand each detail of programming. Performance on a task which investigates only what individuals can produce, or even what they understand, therefore, does not accurately reflect the level of programming which is commonly utilised in the SUNRISE classroom.

1 The mathematics teacher observed that the tendency of the students in the SUNRISE project to approximate, provide answers that are not accurate, etc. in mathematics and science is stronger than he has experienced in other classes he has taught.

8

Gender Differences

This chapter presents a perspective about gender differences in relation to learning with computers, attempts to analyse ways in which technology is viewed and looks at implications of such views for how the computer is incorporated into the educational setting. It will be argued that in addition to the problem of equity of access to the hardware, girls are often not given appropriate support and contexts for learning about and with computers.

Three lines of converging arguments will be examined: (1) Educational computing is commonly identified with the domains of mathematics and science. This classification has overt and covert implications of how they are incorporated into schools and viewed by students and society as a whole. (2) For a number of years there has been widespread concern about sex-related learning differences in science and mathematics, and it is not surprising to find these differences emerging in the area of computing. A large body of research investigating these problems in science and mathematics has taken into consideration attitudes, interests and achievement, career statistics, and analyses of social processes in classrooms. (3) Quite a number of studies have been conducted into the processes of children's learning and the use of computers in education. Some of these studies (cf. Hoyles, 1988; Pea & Sheingold, 1987) are providing us with an interesting body of knowledge about patterns of sex differences in learning with computers. These findings will be reviewed with respect to the perspective developed here, and related to the empirical observations made in the SUNRISE classrooms at Coombabah.

EQUAL OPPORTUNITY

The issue of equity of learning about and access to computers is an important topic for educators. A common concern is that all students should have equal opportunity and appropriate support for acquiring competence with technology. These concerns derive from (a) the belief that, as in the future many careers will require competence with computers, knowledge of this technology will be a source of power, and (b) the fact that currently there are differences among groups of people in their access to bodies of information which may be exacerbated by unequal opportunities for becoming familiar with the technology.

Two major dimensions of such differences are SES and gender. With respect to the latter, if current projections are accurate, girls are likely to learn less about computers than boys, and have less ability to control this increasingly important cultural tool (e.g. Equal Opportunities Commission, 1983; Hawkins, 1987; Moont, 1984).

In Australia, Chambers & Clarke (1987) report a cumulative negative effect for students from disadvantaged groups following class computing experience. They defined disadvantage in terms of gender, scholastic ability, SES and ethnic background and showed that students' perceptions of control of the learning process are actually greater in computer experiences which occur out of school than in the classroom.

A meta-analysis of investigations into gender differences in attitudes, achievement and use of computers (Hattie & Fitzgerald, 1988) showed that although there were no significant differences between boys and girls of primary school age in their use of and attitudes towards computers, marked gender differences became evident as students progressed through secondary school. These differences could be the result of cumulative differential experience and opportunity, but Hattie (1988) attributed them to differences in perceived *control* over process and product.

> The individual with control *expects* a less aversive outcome than the individual without control, and there is also a desire to minimize the maximum danger to themselves. (Hattie, 1988, p. 7)

> A person who has control over an aversive event ensures having a lower maximum danger than a person without control. This is because a person with control attributes the cause of relief to a stable internal source - his or her own response - whereas a person without control attributes relief to a less stable, more external source. (Miller, 1980, p. 80)

The suggestion is that girls rather than boys have a sense that they are not so much in control when they work with computers. Our society may still influence many young girls in such a way that they come to believe that compliance, negotiation and the avoidance of risk-taking are desirable attributes for women. These very attributes are not helpful to those who work with computers.

In our empirical study no gender differences were observed in novices after 2 months of learning with laptops. However, gender differences occurred and increased over the school year, and they were larger in the more experienced group, i.e. Year 7. Boys and girls did not differ in their perceptions of *control* over the technology or their enthusiasm -- rather, with time, the girls perceived themselves to be having, and were observed to have, less input within the classroom, communicated less with the teachers and were less involved in the general planning. This may have led to a lowering of self-image in a number of the girls. The strong gender stereotype held by about a one quarter of the boys at Coombabah that computers are generally more important for men than women, and that as a rule boys are better at writing procedures than girls, is likely to have

had its effect on the girls themselves and on the classroom climate more generally.

A number of researchers have found gender differences in attitudes to computer games and suggest that these differences are due to the types of commercially available games. Keisler, Sproull & Eceles (1988) claim that recreational and educational software is designed with males in mind, and these researchers suggest that different types of games may have to be developed for girls. Clarke (1989) stresses the importance of games as a means of engaging girls in independent learning with computers. If playing with computers is important for later skill development, then game design may be an important educational issue. In the Coombabah classrooms girls reported not to be interested in using the computer for game playing. None of the girls mentioned making up games as a favourite computing activity but 25% of the boys did.

Hoyles (1988) argues that for girls the computer is a tool, rather than a sophisticated technical toy. She suggests that 'focused courses in which the utility and power of the computer are displayed, seem to provide a more direct route to greater participation by girls' (p. 10). Our empirical data certainly support this view. More girls than boys enjoy programming-related activities better than subject-oriented activities, and a larger proportion of girls than boys would use the computer for their work wherever possible, in preference to using paper and pencil. Crawford (1988) reports equal enthusiasm by girls and boys in their first year of schooling for creative play with Logo. However, she found substantial gender differences in the forms and outcomes of undirected computing activity.

COMPUTING AS A TOPIC AND AS A TOOL

Interest and achievement in the areas of science, mathematics and technology have generally been shown to be linked to gender in educational and work environments. As computers continue to play an ever-expanding role in people's lives, their conceptual connection with the disciplines of mathematics and science has important implications for learning.

While computers are used for many other purposes, their computational properties as applied to science and engineering tasks are particularly salient. Mawby, Clement, Pea & Hawkins (1984) found that 8 to 12-year-old children think of computers as particularly related to mathematics and science tasks. The view of computing as a topic subsumed under science/mathematics has serious educational consequences for girls, because these subject domains have generally been linked with activities and careers that have long been dominated by males. Thus, computers typically enter the classroom with an aura of sex-related inequities which affect both learners and teachers in subtle and not so subtle ways.

Conversely, one might view educational computing as the introduction of tools which can be adapted to a wide variety of purposes in all subject areas, i.e. language, art, music, information gathering, and administration, in addition to their time-honoured use in mathematics, science and technical subjects. The conception of the computer as a universal tool which can aid the acquisition of knowledge and skills in a number of ways and areas would serve to broaden its category membership. The interpretation of the role of technology provided in educational settings is central to how girls and women assess its relevance to their own learning.

A number of studies documenting gender differences in the use of computers are finding that boys tend to be more interested in and make more use of the equipment than girls (e.g. Hess & Miura, 1983; Lockheed, Nielsen & Stone, 1983), particularly for functions such as programming (e.g. Becker, 1983-84). This is likely to be a result of limited and competitive access to computers. In a setting such as the SUNRISE classrooms at Coombabah, where all students have their own laptop computer, girls were as interested in programming as boys, and often more so.

As has been the case with mathematics (e.g. Burton, 1979; Minuchin & Shapiro, 1983; Osen, 1974) parents tend to be more supportive of boys learning in this area than girls (e.g. Miura & Hess, 1983). Examination of the literature analysing sex differences in the areas of mathematics and science offers a perspective on the assumptions that underlie these differences for computers. It is important to look at these differences in the context of societal beliefs and social conditions as well as the factors that appear to mediate their developmental course, particularly in schools.

DIFFERENCES IN PARTICIPATION

There is a perception within our community that men participate in computer-based activities far more than women. When women are portrayed in high technology publicity material it is often as the operator *sitting* at the keyboard (i.e. in a subservient, secretarial role) with an authoritative male in a business suit (i.e. the busy executive) *standing* behind and pointing to the screen. Why is there this perceived disparity in the use of computers between the sexes? Are men simply more able in dealing with the technology, either as a consequence of differences in cognitive structures and processes, or as a consequence of educational exposure?

Attitudes towards Male and Female Computer Users

The strength of perceived differences in men and women computer operators was demonstrated in a nicely designed study by Siann, Durndell, Macleod & Glissow

(1988), which investigated student attitudes towards men and women working with computers. Almost a thousand tertiary students were asked to read a short description of an imaginary person and then completed a questionnaire seeking to determine perceived personal attributes of the individual. Attributes tapped by the questionnaire included self-reliance, sympathetic personality, personal adjustment, ambition, approachability, competitiveness, likeability, seriousness, and so on. The description of the individual in question outlined a computer science student who owns a home computer, and is aiming for a career in computer design with ambitions in the direction of higher management. Leisure interests were also described. Two descriptions were used in the experiment, and half the participants read each of these. The only difference between the two descriptions was that in one the individual was someone named Kevin, while the other was named Karen.

The results of this study were not as predicted. There were no differences between the attitudes of the males and females completing the questionnaire. Karen was seen as being more self-reliant, more fun to be with, more independent, more approachable, more likeable, more sympathetic, better adjusted personally, more popular, etc. than Kevin by both male and female students. The overall picture provided of the person Kevin/Karen was one of a competent budding professional with a generally positive personality, but Karen was seen as having more of these attributes than Kevin. If computing is a predominantly male domain, then the opposite of these results might have been expected. For example, Karen might have been seen as being more aggressive in order to succeed in this alien domain, but this was not the case. The results might be explained by the type of women who enter computing being atypical of the sex stereotype. The results of this study certainly support the rejection of ideas that women in technology are stereotyped negatively, at least in the group of students sampled by the survey.

Unfortunately, the above described study is not typical of the general state of affairs. Culley (1988) and others provide a gloomy picture of the participation of girls in computing activities, except in girls' schools. A postal survey of several hundred schools in the UK found that girls showed little interest in computing. In optional activities such as computer clubs the girls accounted for less than 10%. Furthermore:

> Computer rooms in most schools were regarded as male territory and girls report being made to feel very uncomfortable by the attitudes and behaviour of boys. Several schools had recognised this problem and responded by establishing certain times as *girls only*. Such schemes were only partly successful, however. The tendency was for the open sessions to become effectively the boys' sessions and thus reduce even further the access of girls to computers. In one school the open sessions were overseen by a male computer teacher, while the girls only session was staffed by a female who had no computing expertise. (Culley, 1988, p. 4)

This suggests that, at least in some schools, girls might get less time on the machines, and that they could be getting less expert help. Culley observed a

tendency of boys to dominate in classroom discussions about computing, that they tended to direct more questions to the teacher, while the girls tended to be sitting back.

The same is obvious in the SUNRISE classrooms at Coombabah. Throughout the year, in discussions about programming procedures, the boys clearly dominated. A particularly obvious example occurred at the beginning of Term 4 in 1991 during a lengthy brainstorming session on *worlds*, where twice as many boys as girls were observed to volunteer answers. The only general discussions in which boys did not dominate was on the topic of *ways of learning spelling*. To this discussion boys and girls contributed to similar extents. A second problem which the Coombabah data revealed is that girls tend not to seek the teachers' assistance when they experience difficulties in their computing. Only a few girls would turn to the teacher, while the majority of boys did. It must be noted that this tendency of the girls to turn to friends and family rather than to the teacher was not restricted to computing, but also occurred in areas such as reading and spelling.

In the girls' schools sampled by Culley (1988) a different picture emerges. The girls in these schools were enthusiastic about computing, as indicated by high levels of participation in computing options and computer clubs. Culley concluded that the most likely reason for such a difference in the involvement of the girls between the two types of schools lay in the organisation of the teaching with and without computers.

Culley (1988), Hughes, Brackenridge & Macleod (1987) and other surveys in the UK found that fewer girls than boys have access to computers at home. At Coombabah 50% of boys and 50% of girls in both Years 6 and 7 were found to have access to family computers at home, and of course all the children in the sample have a laptop for their exclusive personal use in and out of school. No significant differences in computer awareness, computing skills and attitudes to computing were evident between students whose families own a computer and those who do not, nor were there gender differences.

Interaction between Male and Female Computer Users

Jackson, Fletcher & Messer (1986) found that teachers preferred to organise mixed gender groups rather than single gender groups, but Siann & Macleod (1986) found that in mixed gender pairs the boys were socially dominant and that the girls were less motivated and also tended to be less successful in a Logo programming exercise. When considering this in combination with the results of the Hughes & Greenhough (1989) study, which also found that girls were at a disadvantage in a Logo-type exercise (i.e. when asked to attempt to move a Logo turtle around a short track as fast as possible), one would conclude that there might be a classroom management problem involved, in that particular group combinations might have been preferred by teachers. Other studies (e.g.

Finlayson, 1984; Webb, 1984; Light & Colbourn, 1987) found that girls were as successful as boys, if not more so. One explanation of these discrepant findings might be that the Logo-type task may not have given an estimate of programming potential so much as being a measure of spatial ability. The children in Hughes & Greenhough's (1989) and Siann & Macleod's (1986) studies may have had difficulty in relating the position of the turtle with the spatial-perceptual-logical manipulations necessary to obtain the desired position. The gender differences associated with spatial ability are well documented. They are also discussed by Siann, Durndell, Macleod & Glissow (1988), with much of the evidence pointing to a superiority in such tasks for males. In the Coombabah sample students chose their own work and group partners. Most of the time the students chose to work with those of their own gender. There was evidence of gender differences in spatial ability.

Performance by Male and Female Computer Users

After reviewing the literature one is tempted to ask the question: Is it possible that girls are simply less able when it comes to working with computers? In their Logo programming exercise, Hughes & Greenhough (1989) found that pairs of girls worked less successfully than pairs of boys or mixed pairs. This study is one of the few to have found that girls work at a disadvantage in computer-based tasks, and Underwood & Underwood (1990) point out that there is good evidence to suggest that there are no gender differences in ability to use computers. Hughes & Greenhough attribute their findings to differences in attitude rather than in ability. But then, attitudes and abilities are related, they interact and transact. For the majority of Year 6 and Year 7 girls our empirical study showed no differences in the abilities of boys and girls in computing, nor in their enthusiasm. However, the number of girls who fell behind the rest of their class over time was larger than the number of boys in this position.

In an evaluative study of the benefits of Logo programming reported by Finlayson (1984), a class of Logo users was compared with non-programmers on some tests of mathematical ability. Differences in performance on the mathematical tests were also analysed for sex differences. Although the 11-year-old boys spent more of their free time than girls using computers, there was no difference between the groups. The boys often chose to use computers rather than be involved in other class activities during their free time, and the girls tended to avoid the machines except at set times, even when the boys were not present. The boys spent half as much time again than girls using computers, yet the girls performed just as well in the pencil and paper tests which were regarded as sensitive to Logo programming exercises.

It should be noted that the gender differences observed by Culley (1988) were not generally associated with ability in computing. As was noted above, Underwood & Underwood (1990) described a long series of studies, conducted in

the UK, and found no gender differences in computer-based learning and programming tasks. They point out that the results which suggest that girls are less able with Logo programming tasks are an exception. They found that girls and boys perform at the same level in a variety of computer-related tasks. They also found that girls are less likely to hold stereotyped attitudes about gender differences in ability, a finding supported strongly by the Coombabah study. The latter point supports Eastman & Krendl (1987), who found that children in a middle school science class learnt how to access an electronic encyclopaedia and collect materials to support their writing projects. The study found no differences in success of computer operation between boys and girls but there were differences in attitudes. In particular boys were more likely to think that computers were for boys and that boys were more able users of computers. The girls were less likely to hold stereotyped views, but there were no significant differences in the attitudes of boys and girls towards gender roles.

In the Coombabah study boys and girls did not differ in the view that learning computing is important for both males and females. However, nearly a quarter of the boys, and only one girl, believe that boys are actually more able programmers than girls.

The weight of evidence goes against the notion of girls being less able in computing, and the safest interpretation of the differences observed with moving the turtle (Siann & Macleod, 1986; Hughes & Greenhough, 1989) might be that these differences were due either to the constrained nature of the available responses (correctness was very constrained), or the task's providing measures more representative of spatial ability than of Logo programming ability. Obviously, schools and teachers need to develop strategies to ensure that girls participate more fully in opportunities available to them to acquire computing skills. As discussed later in this chapter, the differences observed in the SUNRISE classrooms at Coombabah are likely to reflect gender stereotyping on the part of the boys.

As many other researchers have argued, students' perceptions of subjects, and their behaviour in relation to them, are effects of wider social forces as well as the result of classroom organisation. The career aspirations of pupils significantly influence subject choice at school, and there is evidence (cf. Culley, 1988) that fewer girls are attracted to careers in computing. The research found that career aspirations of students were highly differentiated according to gender and that careers in computing played very little part in girls' hopes for their future occupational roles. Working with computers was a significant aspiration for a large number of boys though.

Interviews reported in the literature and our observations at Coombabah revealed that teachers tend to see boys as more noisy and demanding than girls. Boys are not always sufficiently motivated towards written presentation of their work but they were frequently regarded as more interested in computing, and in the latter area seen as more rewarding to teach than girls. Even when girls did all that was required by the teacher, followed instructions carefully and presented

their work well, they were still seen by some teachers as less interested and as having less *flair* for computing.

It is common for teachers and other educationists to play down the importance of processes within the school in determining the attitudes of students generally and the existence of sex stereotyping in particular. Several teachers and principals interviewed during the research sought to explain gender differences in school subject choices by reference to social assumptions and processes which were seen to have their origin outside the school in family socialisation, media images, and so on. There is, it was implied, little that schools can do against the power of such outside influences. It is not suggested here that the practices of schools and teachers are the only factors involved in gender differences in schools in general and sex-typing of computing in particular. They are, however, of high significance. Schools are part of society, and what goes on in school interacts with *outside* forces and is itself part of the creation of social relations, attitudes and assumptions.

One of the female teachers in the SUNRISE project at Coombabah has not really come to terms with the role of computing in the classroom and in society as a whole. She expressed to the author her belief that there are genetic reasons for the differential attitudes and success of males and females with respect to computing. This view, even if not expressed overtly, is likely to assist the development of gender stereotype in her students and have detrimental effects on the self-image of the girls in the class.

Factors Contributing to the Development of Differential Interest and Achievement

Contrary to the findings of our study at Coombabah, Steinkamp & Maehr (1984), after testing effect size for published findings, confirmed that sex-related differences in educational achievement and motivation appear with the introduction of computers into the classroom. Sex-related differences in educational achievement have been explained in terms of (biologically or socially acquired) differences in abilities, cognitive style, and possession of knowledge, in addition to societal expectations and stereotyped sex roles. The psychological literature has traditionally claimed that girls excel at verbal tasks while boys are better at spatial and abstract tasks (Haertel, 1978; Maier & Casselman, 1970). Gender-related differences in motivation have been explained by assuming a relationship between attitudes towards science and mathematics and self-concept (Eastman & Agostino, 1986; Handley & Morse, 1984). Although gender-related differences may apply to new technologies, few studies have rigorously examined these differences in relation to communication media.

Maccoby & Jacklin (1974) reviewed the psychological literature on gender differences, Eakins & Eakins (1978) summarised gender differences in human communication, Lawton & Gerschner (1982) dealt with attitudes to computers and computerised instruction, but none of these reported studies relating gender to new technologies. Research into children's cognitive skills shows computer-related attitudes and abilities can be expected to vary between the sexes after about the age of six years (e.g. Kirchner, Martin & Johnson; Paisley, 1983; Williams & Williams, 1984). Recent educational applications of computers support this claim, typically asserting differences in achievement (operating skills) between females and males (Paisley & Chen, 1982; Rice, 1984; Williams & Williams, 1984). Based on these studies researchers should expect boys to outperform girls on computers.

However, there are some problems with these findings: as noted above, the particular task may be the key indicator of achievement. In studies showing gender-related differences in motivational attitudes toward computers, girls expressed less desire to learn to use computers than boys, and were less adept, at least initially (e.g. Corno & Mandinach, 1978; Lepper, 1982; Williams & Williams, 1984). Such studies typically focused on computer programming or using the computer for mathematical drill and practice. Gender differences in attitudes towards mathematics and sciences (traditionally more negative for girls) and towards libraries and writing (traditionally more positive for girls) may confound findings about computer performance if the kind of task influences computer learning (Anderson, Klassen, Krohn & Smith-Cunnien, 1983; Erickson & Erickson, 1984). It has been assumed that students associate computing talent with science and mathematics, and hold stereotyped attitudes on girls' ability and interest in mathematics and computer operations (e.g. Deboer, 1984; Handley & Morse, 1984; Levin & Fowler, 1984; Scott, 1984; Steinkamp & Maehr, 1984). In general, girls have been shown to possess more negative attitudes toward mathematics and science (e.g. Fox, 1977) and towards computers in particular (e.g. Williams & Williams, 1984; Winkle & Mathews, 1982). Extrapolating from research into gender-related attitudes and skills, then, researchers could expect boys to demonstrate greater motivation to learn to use computers and to master operating skills more rapidly and thoroughly than girls. This expectation was not fulfilled in our empirical study, which showed equal motivation, enthusiasm and skills for boys and girls.

Another, perhaps more important, question is whether the nature of the tasks posed in learning with computers supports the engagement of girls as well as that of boys. Research has shown that particular features of the learning of tasks as interpreted by individuals may be an important factor in the development of gender differences. For example, achievement orientations might differ for the boys and girls in different subject domains. Girls and boys tend to interpret negative feedback differently; girls are more likely to attribute difficulty in problem solving to their own lack of ability, whereas boys are more likely to attribute failure to other situational factors, a finding strongly supported in the Coombabah study.

Mathematics tasks are commonly presented in such a way that the occurrence and salience of failure is greater than for language or social science tasks (cf.

Brush, 1980), i.e. the solution of a mathematics problem tends to be either correct or wrong, and the correct solution to a task illustrating a mathematical concept that is new to the student is often preceded by a series of wrong answers. In contrast, many language tasks are interpretative (e.g. writing an essay) and therefore subject to more flexible evaluation which, in turn, may lead to further development of the ideas expressed. In this respect computer programming has probably more in common with experiences of success and failure in mathematics than in language tasks. Frequent encounter of failure with procedure writing tasks, i.e. finding that the procedures do not *work*, may thus be interpreted differently by girls and boys with respect to their self-perceived abilities.

Lenney (1977) offers considerable support for the argument that men and women react to achievement situations differently. Her analysis of adult performances indicate that women's self-confidence seems to be affected by specific task characteristics, the kind and quality of feedback offered, and the degree to which competition and evaluation play a part. For example, she presents evidence that women are more likely to express confidence in tasks that feature social as opposed to intellectual skills. Women also appear to be less confident than men in situations where there is little or ambiguous feedback. She concludes that, as a general attribute, women may be no less confident in their ability to achieve than men are, but they may be more sensitive to particular characteristics of an achievement situation in assessing their own competence.

Parsons, Kaczala & Muce (1982) were interested in understanding which social processes in classrooms might give rise to differential feedback for boys and girls, and thus to differential expectations and self-confidence in mathematics. Classrooms were observed to differ in the amount and kind of feedback given by teachers to boys and girls of low and high mathematics achievement. They found that boys and girls have equivalent achievement expectations when praise and criticism are equally distributed across gender and teacher groups. However, the social processes in the classrooms can influence children's expectations for themselves: girls were shown to have lower expectations for their own performance in classrooms where they are treated differently from boys. There is some evidence that at least some of the girls in the Coombabah classrooms were treated differently from the boys, even though this differential treatment may have been self-induced by the fact that the girls chose to seek the teachers' advice less frequently than their male peers.

Another question is whether particular aspects of the more general culture (media, parents, educational authorities) give messages about gender-appropriate interests that are perceived differently by boys and girls. As was noted above, advertising of computers is overwhelmingly male oriented. The majority of products advertised as well as jobs in the area of computing technology are directed towards a male audience.

Educational Programs

A variety of promising educational programs aimed at changing the self-perceptions of girls with respect to science and mathematics as necessary skills for future careers have been developed in recent years (cf. Brush, 1980; Kreinberg, 1981; Hawkins, 1987). Similar programs are needed in the area of computing. However, since it is often difficult for young students to think as far ahead as careers in adulthood, this approach alone is unlikely to be adequate.

It is important to help students recognise that computing, as well as mathematics and science knowledge, are tools which they can use in their everyday lives. In many school settings relatively little effort is made to adapt the learning content to children's interests and orientations. Brush (1980) suggests that mathematics teachers can make classes more enjoyable for students by developing mathematics tasks that emphasise creativity and interpretation rather than mere success and failure. She offers the example of a teacher who incorporates geometry skills into a project to design the layout of a room. By the same token, girls may benefit from and feel more involved in computer programming and other computer-related learning experiences that are relevant to their current interests and circumstances.

PATTERNS OF DIFFERENCES

As noted above, most of the work on gender differences in science and mathematics discusses these issues holistically rather than by focusing on the areas, skills, applications and the contexts that may engage individuals differentially. If computing is viewed as a subject area in its own right, it is likely that differences in interests and achievements of boys and girls are similarly analysed in global terms. The literature already contains reports of significant overall gender differences in computing which in no way address the characteristics of the particular situations where the differences are found. Lenney (1977) and others have shown that aspects of the work context are very important in understanding the appearance of gender differences in achievement. Therefore, an examination of the pattern of differences between male and female students in the use of the computer as a tool in and out of school is of utmost importance.

The Center for Children and Technology (CCT), Bank Street College, New York, conducted an in-depth survey of three geographically disparate school districts and found that gender was the most obvious factor affecting differential uses of computers at all year levels and sites (Sheingold, Kane & Endreweit, 1983). Findings pertaining to gender differences emerging in this study, conducted mainly as case studies with 8 to 9 and 11 to 12-year-old children, will be discussed here under three headings: programming, word processing and mathematics and science software.

Programming An increasingly common use of personal computers at all school levels is for programming. Investigators from the CCT conducted a study over a period of two years to investigate children's learning of the programming language Logo. These studies addressed two issues: (a) the cognitive aspects of learning to program, and whether knowledge of programming concepts would be generalised to other problem solving situations (Pea, Hawkins, Clement & Mawby, 1984); and (b) the social and organisational aspects of incorporating personal computers into classroom settings (Hawkins, Sheingold, Gearhart & Berger, 1982). Measures used in this study included interviews and tasks which examined students' understanding of commands and concepts, the complexity of their programs and their monitoring of ongoing work. In general a clear trend emerged for boys to perform better than girls on all the tasks. The girls showed less interest than the boys and developed less facility with Logo -- finding which was certainly not supported in our study at Coombabah.

At the end of the year, all children were given the programming knowledge assessment which consisted of three parts: (1) knowledge of individual programming commands (definition and use); (2) ability to write a variety of short programs to execute specified goals; and (3) ability to debug programs containing different classes of errors. (For a detailed description of the tasks, scoring procedures and findings, see Pea, Hawkins, Clement & Mawby, 1984.) In both age groups boys performed significantly better on all measures of programming expertise, and in general showed more enthusiasm for the work and spent more time programming (mean: boys 34 hours, girls 22 hours, $p<.01$).

With respect to knowledge of programming commands, the mean score for boys was 47.2 and for girls 25.1 ($p<.01$). There were also marked sex differences in program composition skills. In this part of the assessment students were asked to write lines of code using increasingly sophisticated programming concepts. Older children were more skilled than younger children, and the boys in each age group displayed more skill than the girls. In this analysis, a student's efforts in each of seven subtasks was classified into one of three categories: (a) correct, (b) partially correct (lines of code were correct but the child failed to organise them procedurally or to return the *object* which executed the program to its starting position); and (c) wrong or no attempted solution. The younger boys wrote correct or partially correct programs in 36% of the cases, younger girls only 6%; older boys were correct or partially correct 70% of the time, older girls only 26% of the time.

Similarly, boys displayed more programming skill in the third component of the assessment, namely the debugging of faulty programs: the mean score for younger boys was 31.1 and 19.9 for younger girls; the mean score for older boys was 48.9 and 17.4 for older girls.

At the end of the year the children completed a questionnaire which, among other items, asked them to nominate two class members who were, in their judgment *experts* in computing. Three boys were overwhelmingly selected by the older children and two boys by the younger ones. The teachers at the end of the first year of the study also judged the performance of the four boys as best.

When the teachers assessed the first year's work of computer programming, they reported dissatisfaction with the progress of most children and expressed particular concern about the apparent sex differences. As preparation for the second year, the teachers reorganised their presentation of the material so as to better support children's learning (e.g. presenting a more structured sequence of concepts, development of project ideas). During both years, finding functional goals for their work as they learnt Logo was a continuing problem for the students. Many did not have a clear understanding of how to adapt computer programming to projects in which they were interested. Over the course of the second year, the teachers tried to spend more time with the girls and to devise projects (such as programming word games) that might better accommodate their interests. They certainly helped them to use new skills. However, by the end of the second year, the teachers reported that they continued to see gender differences in the amount of interest in and commitment to programming tasks.

In contrast to the first year of the study, the teacher of the younger children identified four girls (one *outstanding*) and two boys as experts. However, of the 11 children in this class judged to be proficient, 10 were boys. Sex differences were particularly striking among the older children (11 and 12-year-olds). The teacher judged six boys and one girl to be experts, and seven boys and three girls to be proficient.

It is important to note that these sex differences did not appear across the board: as noted above, four girls in the second year class developed considerable expertise. There were individual girls in each class who displayed a lot of interest, performed well, and were judged by both teachers and peers to be competent with computers. The expert girls tended to be competent in all school subjects. This overall competence was not always true of boys, some of whom had previously shown little interest or competence in school but who *blossomed* when they started working with computers. As described in Chapter 7, the same pattern evolved in the Coombabah study.

The studies examining the development of programming skill in the younger children showed strong differences between boys and girls in levels of interest and achievement. This is particularly striking in light of the teachers' sensitivity in these studies to the problem and their efforts to give special help to the girls. As was noted above, in the Coombabah study the teachers did not set out to give extra help to the girls. To the contrary, when the decreasing motivation and increasing falling behind of some of the Year 7 girls was pointed out to one of the teachers by the researcher, the teacher explained that nothing could be done.

Word Processing Published research and our empirical observations showed that boys and girls are about equally involved in word processing (Pea & Kurland, 1987b). Word processing appears to invite more collaborative writing among students than traditional methods. In the literature there is some indication

that collaboration may be preferred by more girls than boys (cf. Hawkins, 1987). Our observations showed no gender differences but rather that the high achieving students tended to prefer working by themselves. While it is unclear whether students wrote *differently* as a result of their experience with the word processor, it was noted that many write *more*. No sex differences were found in our own and other studies in the use of and achievement in word processing. When given a choice of activity the majority of the Coombabah students, both boys and girls, selected programming. The published literature finds this to be true for boys rather than girls.

Mathematics and Science Software Another project of the CCT at Bank Street College, New York, was concerned with the use of software in science and mathematics by students in Year 4 through to Year 6 classrooms. Three pieces of software were developed and tested, both with individual children and in larger groups (Char, Hawkins, Wootten, Sheingold & Roberts, 1983). The software was designed to make use of the unique and powerful features of the computer and to model ways in which tools are actually used by adults in their work. This research is part of a larger project, the US *Project in Science and Mathematics Education*, the aim of which was to produce an integrated set of software and materials for classrooms, television and videodisc. One mandate of the project was to encourage girls to develop an interest in science and mathematics. The three pieces of software included: (a) a tool to gather data about physics phenomena (temperature, light, sound) and to display these measurements in various types of graphic formats; (b) a simulation to introduce principles of navigation and the geometry involved; and (c) a series of games designed to introduce children to programming concepts in Logo. Since these programs fit into the existing mathematics/science curricula of elementary classrooms, the pattern of findings concerning girls' participation are especially interesting.

Differences between girls and boys were most notable for one of the three pieces of software -- the tool used to gather and display data. Boys tended to make more use of this software than girls, often working in groups, with girls either not interested or watching from afar. The other two pieces of software were no less technical or mathematical, yet there were few apparent sex differences in their use and appeal. For example, the triangulation principles introduced in the simulation software were complex mathematical concepts. In the case of the programming games, girls were more likely to report that they liked the software (83%) than boys were (64%). The simulation program appealed equally to both sexes, and there were no appreciable differences in students' questions to comprehension questions.

One might speculate about two features of the software that contributed to its appeal for girls. (1) Learning experiences with these two pieces of software tended to be collaborative enterprises. The simulation game was designed in such a way that students were required to play cooperatively. Teachers also chose to organise the programming games as collaborative work between pairs of students. (2) The goals of the software were less explicitly scientific than was the case for the data-gathering tool. The latter was introduced as part of the science curriculum for conducting experiments using scientific method. The experimental orientation was problematic both for the teachers, who had little training in science and were not at ease with tasks requiring scientific experimentation, and for the girls, who expressed little interest in this kind of task.

In contrast, the goals of the other two software pieces were less directly tied to the traditional mathematics/science curricula. Mathematics/science concepts were embedded as useful tools in achieving the goals of the games (rescuing a trapped whale, finding locations on a map). The teachers were less likely to incorporate these pieces of software directly into the mathematics/science lesson, but rather to use them as independent learning units. Thus the pattern of gender differences in these studies is interesting in that the differences appear to be related to the particular use of the computer, and the way the use is organised and supported in the classroom.

SUMMARY OF SIGNIFICANT GENDER DIFFERENCES IN THE COOMBABAH STUDY

Gender Differences Develop Over Time

No gender differences were observed with respect to *computer awareness*, feelings and attitudes about computers and computing in the responses of Year 6 students to the questionnaires administered in April. However, at that stage some significant gender differences were evident in Year 7 students. The latter students had had their computers for one year longer than the Year 6 students. It would appear that gender differences are not generic but develop with computer familiarity and use. This view is further supported by the finding of highly significant gender differences in the responses to the November questionnaires in both Year 6 and Year 7. These are reported for the total sample (n=115) and for Year 7 (n=59) in the next sections of this chapter. None of the questionnaire items yielded significant gender differences for Year 6 only.

Significant Differences in the Total Sample

The two most highly significant differences between boys and girls in both classes are:

1 When *stuck* with a procedure or any other aspect of a learning task, the boys tend to ask the teacher for help while very few girls do. No matter what the subject area is in which they encounter a problem, the girls turn to a friend for help. Second most frequently they ask members of their family, while the boys

second choice is the class *expert*. This may explain the fact that some girls have adapted less well to computing. It would be interesting to whether this phenomenon would occur in other schools or whether it is a consequence of the particular relationship the teachers at Coombabah have with their female students. As was noted previously, at least one of the female teachers in the Coombabah project holds quite a strong view that the *female brain* reacts differently to learning with computers from the *male brain*, and that thus females are *not made* for computing. Unconsciously, she may have projected this view to her students. On the other hand, one other female teacher is extremely enthusiastic, and a very competent and creative programmer. One might have expected that (although she secretly prefers teaching boys) this teacher might have provided a strong role model for the girls and thus compensated for the attitude of her colleague.

2 When asked to name their favourite activity with the computer, 25% of the boys but none of the girls say: *making up games*. It is possible that the games taught in class were more suited to the interest of boys. On the other hand, learning to make games may have been yet another activity learnt in more informal communications between teachers and groups of students which tended not to include girls.

Classroom observation has shown that the boys are more actively involved in group brainstorming exercises in the classroom. The teachers call for their views more often than for those of the girls. A reason for the latter might well be that the girls volunteer fewer views and suggestions.

Twice as many boys (70%) as girls believe that learning how to use computers is as important as learning mathematics, reading, spelling, etc. ($p < .007$) and that one day computers will replace teachers ($p < .003$). 34% of the boys but only 12% of the girls believe that computers can teach better than teachers ($p < .01$), which may explain why the girls tend not to ask help from their teachers; and 88% of the boys but only 73% of the girls feel that computers are better than textbooks ($p < .05$). And yet, only 46% of the boys and 27% of the girls try to find books about computers or computing in the library ($p < .04$); for Year 7 the proportion is 45% of the boys and 4% of the girls ($p < .0003$). 56% of the boys and 81% of the girls discuss computers and computing with members of their family at home ($p < .005$).

An interesting finding relating to these two issues is that 62% of the Year 7 girls but only 23% of the boys believe that students working with their own computers require less attention from the teacher than students in a conventional classroom. This finding might be explained by the observed tendency of these girls not to ask the teacher for help.

56% of the boys and 32% of the girls feel that using computers *never* makes schoolwork more difficult, but 79% of the boys and 96% of the girls believe computing to be very timeconsuming. 53% of the boys but only 32% of the girls believe that students who have become accustomed to using a personal computer will find it difficult to work without one.

When asked to name the subjects in which they like using the computer most, 85% of the boys (59% of the girls) named mathematics as their first preference, and 97% of the boys (81% of the girls) named social studies as one of their two top preferences. In response to the question *Which subjects do you not like as much as you used to, now that you are using a computer?* 30% of the girls (12% of the boys) mentioned mathematics and 34% of the boys (11% of the girls) mentioned science. Only 34% of the boys and 45% of the girls stated that they like all subjects as well as in the past or better.

Significant Gender Differences in the Year 7 Group

Only 10% of the boys but 32% of the girls in Year 7 ($p < .04$) are still reporting in November (i.e. after their second year in the SUNRISE classroom) that using their laptop makes them feel nervous. 66% of the boys but only 39% of the girls ($p < .05$) are reading books about computing in November. However, 90% of the boys and 62% of the girls reported that they had, at some stage, read such books ($p < .04$).

76% of the boys and only 50% of the girls ($p < .05$) are finding that using computers makes mathematics learning easier. However, 50% of the girls and only 14% of the boys believe that they are *not the type to do well with computers*. Only 10% of the boys but 32% of the girls still cannot understand how some people get so much enjoyment out of working with computers.

Gender Stereotype

Gender stereotype was investigated on the basis of the students' responses to the following three direct statements, which were interspersed with other statements and questions in the questionnaires:

In general, computers are more important for men than for women.
In general, boys are better than girls at writing procedures.
In general, girls are as good as boys at writing procedures.

There was no difference in the responses of Year 6 and Year 7 students with respect to the first of the above statements. In both classes 14% still believe that, in general, computers are more important for men than for women. However, in the total sample 24% of males and only one female agreed with the statement. Only 6% of students in the Above Average IQ category but 17% of Average and Below Average students agree that computers are in general more important for men than for women. The Above Average students who agreed came from Year

7. 7% of R1 but 26% of R5 students agreed, coming in roughly equal proportions from both classes.

In response to the second statement, 29% of all students believe that in general boys are better than girls at writing procedures, i.e. 24% of Year 6 and 33% of Year 7 students. 51% of all males but only two of the females held this belief.

23% of the students in the Above Average IQ category and 27% of Average but 54% of Below Average students agreed that, in general, boys are better than girls at writing procedures. There were differences between the years in this. 24% of Above Average IQ in Year 6 and 21% in Year 7 agreed with this statement. 21% of the Average IQ group in Year 6 and 31% in Year 7 agreed, while 40% of the Below Average IQ students in Year 6 and 62% in Year 7 agreed.

26% of R1 students and 39% of R5 students agreed that boys are, in general, better than girls in programming. This represents twice as many Year 7 as Year 6 students for both the R1 and R5 categories.

In response to the third statement listed above, 24% of all students showed that they believe that, in general, girls are not as good as boys in programming a computer, i.e. 20% of Year 6 and 28% of Year 7. In fact 44% of all males but only one girl hold this view.

80% of students in the Above Average IQ category, 77% of the Average IQ group but only 61% of Below Average students feel that, in general, girls are as good as boys at writing procedures. This means that between 20 and 30% of the students hold stereotyped views which discriminate against females. In Year 6, 24% in the Above Average, 17% in the Average and 20% in the Below Average IQ group believe that girls are not as good at writing programs as boys. In Year 7 the proportions taking this discriminatory view are 14%, 29% and 50% of Above Average, Average and Below Average students respectively. 30% of R1 and 36 % of R5 students believe that, in general, girls are not as good as boys in programming computers.

It appears that negative gender stereotype with respect to computer programming ability is stronger in boys than in girls. This is not surprising. What is more surprising is that even students of high intelligence are not immune to the fallacies which are so strongly prevalent in our society. The results of this study suggest that students of lower ability and computing achievement are likely to be more prejudiced than students of high intelligence and computing ability, and that older students might be more prejudiced than younger ones.

CONCLUSION

Gender differences in attitudes to computing, and to motivation and computing achievement, develop over time in students who are learning with computers. In situations of guaranteed equal access to computers, gender differences are not evident before the students have had considerable personal experience in computing.

Research concerned with the emergence of gender differences in relation to learning with computers and student achievement indicates that this is a complex and deeply rooted problem which appears to be related to many factors, including the impact of differential societal images, perceived expectations and the expectation of different life goals for boys and girls, the structure of learning tasks, the nature of the feedback in performance situations, the organisation of classroom settings and the overt and covert reactions of teachers to their female students.

Investigations of gender differences typically focus on the general domain, e.g. mathematics, science, computing, etc., where inequalities are apparent. What this research has shown is that it is necessary to look deeper, e.g. to examine the functional uses of the materials in particular situations in order to understand the contexts in which boys and girls express interest and achieve competence.

Computers are tools which can be used for a variety of purposes. However, in the absence of a broader perspective, many schools subsume them under mathematics/science curricula, and thus they take on an existing stigma of sex stereotypes. There are two promising ways to reduce this stigma. Firstly, it is important that computers be used in classrooms as tools which help achieve a variety of goals (e.g. word processing, spreadsheets, databases, music, mathematics, planning of events). The interests and goals of individual students need to be matched by specific computing activities, along with appropriate support for learning about the technology. Secondly, the careful design of software in the areas of mathematics and science, as well as more suitable computer games/simulations, may enable girls to view these subject domains as useful to them personally. Of course, this also requires taking into account both the design of curriculum and the general organisation of learning in the classroom.

PART IV

ASSISTING THE TEACHER

9

Professional Development for Personal Computer Use in the Classroom

This chapter suggests a general framework for the effective professional development of teachers who teach with computers, and raises important issues for discussion. It reports on teacher recommendations regarding the conditions, the content and the organisation of professional development programs. Recommendations are made for special resources and the promotion of computer literacy among teachers more generally.

GENERAL AIMS

The two most important general aims of professional development for teachers teaching students who are learning with computers are likely to be:

1 improving the skills and confidence of individual teachers in computer use: and
2 persuading teachers to explore educational computing and to integrate computing into their teaching practice.

The lack of adequate training and experience with computers in the classroom presents a major source of frustration for teachers. In all states of Australia, the great majority of teachers using computers were either unprepared or inadequately prepared when first confronted with computers in their schools. A solution to this problem appears to be obvious: the provision of pre-service training and ongoing staff development programs. The problem lies not in the solution but in its implementation. Many educators, at all levels, lack the experience and the information which might guide personal decisions regarding the selection of topics for staff development and the organisation of the training.

The literature relating to the implementation of technology in schools contains a number of studies in which researchers collected teachers' recommendations for the content and organisation of training in instructional uses of computers. The

staff development literature which bears on the content and organisation of teacher professional development in instructional computer use was also reviewed. Important ideas concerning both the content and the organisation of preservice education and in-service professional development in teaching students with computers recommended in the literature were snmmarised and discussed with teachers teaching with computers in Victoria and New South Wales (n=67). The preferences and recommendations of these teachers were finally presented to the teachers in the SUNRISE classrooms at Coombabah for their evaluation and comment.

This chapter concludes with a set of recommendations applicable to pre-service education and in-service professional development which is based on the pereeived requirements of a larger group of teachers than those involved in the SUNRISE classrooms at Coombabah, but the latter group's input was weighted more heavily.

THE PROBLEM

Staff development for personal computer use in learning and teaching is being discussed as a major concern not only in English-speaking countries but in others as well. The solution to the problem has been to provide staff development to help implement computing in schools. Concerns about the effectiveness of this solution relate to a number of areas, including:

- not enough teachers can avail themselves of the courses offered;
- the courses tend to be too short in duration;
- the amount of hands-on experience in the topics covered tends to be insufficient in these courses; and
- classroom-based follow-up after the staff development course is not provided.

It is not surprising that even after participation in such courses, teachers feel that their effectiveness in teaching with personal computers often falls short of their expectations.

A number of factors contribute to these limitations in the effectiveness of staff development programs. Firstly, personal computers have entered our schools relatively recently, i.e. during the last decade. Now the number of computers in schools is increasing at an overwhelming rate, in most cases far outstripping the school's or region's ability to prepare teachers to use the new technology.

A second factor is the sheer number of teachers who require training. Although the number of computer-related courses offered in tertiary education institutions is increasing, few of the currently employed teachers received pre-service education in computer use. Even now, only a few schools of education in tertiary institutions have changed their requirements to ensure that every teacher who graduates has acquired a basic competency in educational uses of personal

computers. Moreover, at a time when the need for teachers who are proficient in personal computer use is increasing, the provision of both pre-service and in-service staff development is hampered by decreasing government spending on education. This means that the degree to which the training gap will be filled, and when it will be filled, might depend less on need than on economics.

The third factor limiting effective personal computer use by teachers in classrooms is the lack of knowledge and agreement about the topics and ways of delivering staff development programs. Providers employ different models for staff development, and there appears to be a lack of information about what actually leads to successful implementation and effective instructional uses of personal computers in schools. For example, what content should be covered in such training? Obviously, teachers need to know how to operate the computer, how to load and save instructional programs. But does every teacher need skills in programming, in evaluating software, in integrating computers into regular teaching across the curriculum and in the adaptation and/or development of curriculum materials?

With regard to the design of professional development programs, issues which are yet to be addressed include the following:

- How much training is needed to enable teachers to use personal computers effectively?
- Where should courses be located?
- What incentives could be offered to ensure strong participation?
- How can time release for teachers be managed?
- The shortage of readily available staff development materials, both in printed text and on disk, reflects, in part, the lack of systematically derived empirical evidenee upon which such materials could be based.

A FRAMEWORK FOR STAFF DEVELOPMENT

For the purposes of the present discussion staff development, with regard to learning and teaching with personal computers, is defined as 'the provision of activities designed to advance the knowledge, skills and understanding of teachers in ways which lead to changes in their thinking and classroom behaviour' (Fenstermacher & Berliner, 1983, p. 4).

To place this definition within the context of schools or administrative regions, the assumption may be made that staff development activities could be 'internally proposed or externally imposed, in order to effect compliance, remediate deficiencies, or enrich knowledge and skills of individual teachers or groups of teachers, who may or may not choose to participate in these activities' (Fenstermacher & Berliner, 1983, p. 4).

Together, the above definition and assumption can provide a framework for understanding what might comprise suitable staff development in educational

applications and uses of personal computers, regardless of the specific content of a particular staff development activity.

Staff development activities which fit the above definition would be expected to enhance knowledge, skills and understanding in ways which lead to changes in thought and action with respect to the use of personal computers for learning and teaching. The suitability and value of such staff development activities can be further determined with respect to four important features of the organisational context assumed above. These are reflected in the following questions:

- How was the professional development activity initiated?
- For what purpose?
- Who participates?
- How is participation decided?

On the basis of the features of professional development activities elicited by these questions, it becomes possible to construct a profile of any particular staff development offering which provides a first prediction of whether the activity will serve its intended purposes. Additional criteria will be discussed below.

The literature on staff development suggests that externally imposed activities serving to attain teacher's compliance and requiring them to participate will, all other things being equal, be less valuable than activities which are proposed, at least in part, by teachers who may choose to participate or not, for purposes of enrichment or remediation.

The applicability of the organisational assumptions underlying the above questions can be demonstrated by two contrasting fictitious examples of staff development activities.

Example 1:

A region has placed computers in every primary school. In order to encourage teachers to use them, the regional director arranges for staff development activities in personal computer use for all teachers in the region by employing an outside consultant to conduct two three-hour workshops at each school.

Example 2:

John Smith, a Year 6 teacher, decides to bring his own personal computer to school in order to enrich his students' knowledge of computers and computing. For this purpose he has borrowed some demonstration disks of Logo. Students and their parents were so enthusiastic about the computing that the principal and other teachers decided to try some computing in other classrooms. Eventually a number of personal computers were purchased for the school. By this time, John Smith had become the computer expert at the school and trained interested teachers in ad hoc sessions which were held when two or three colleagues expressed interest and could make time to attend. He helped teachers with new software and with specific problems, and answered questions arising from their particular ways of and aims for using computers.

Example 1 could be characterised as a *top-down* imposition of staff development activity which was externally initiated, and initiated for the purposes of enrichment (or compliance?), with all teachers participating because attendance was mandatory. Example 2 demonstrates a *bottom-up* approach where staff development activities were initiated by teachers themselves who participated voluntarily in order to learn about computers and computing in the classroom.

The organisational factors contrasted in these examples highlight two important issues for the planning of staff development activities. The first issue concerns whether top-down or bottom-up initiation is more effective. There is some evidence in the literature that the latter provides activities that participants will more readily view as valuable contributions to their knowledge, skills and understanding. Another advantage of small-scale staff development activities provided by an experienced colleague is that it can flexibly accommodate the timetables of other teachers, and adapt to their specific questions and needs. Generally, however, successful provision of staff development probably requires a balance between teacher and departmental initiation. With respect to example 2, once computers are available, it would seem reasonable for the region to build on the staff development initiated by teachers such as John Smith and others at the local level, to ensure successful implementation of computing in other schools.

SHOULD PARTICIPATION IN STAFF DEVELOPMENT BE MANDATORY OR VOLUNTARY?

Although the fact that attendance is mandatory does not necessarily diminish the potential value of an activity, voluntary participation seems to lead to more valuable outcomes. Voluntary rather than mandatory participation seems to be a more feasible and reasonable approach to staff development activities relating to computer use in classrooms for the following reasons:

1. Voluntary participants are more likely to view the staff development as enrichment, as something valuable to themselves and their careers.
2. Becoming familiar with and using computers involves a large commitment of time and energy on the part of the teacher, a commitment which is unlikely to arise from involuntary participation.
3. Some teachers have legitimate objections to the use of computers in schools, or in their particular subject areas (perhaps because they feel that suitable software is not yet available) and so choose not to participate on reasonable grounds.

Considering solely the definition and assumptions about staff development by themselves leads to only weak predictions about the potential value of the activities. Clearly, some activities initiated in a top-down manner can be successful, especially if strong efforts are made to enlist teacher support for them.

However, Fenstermacher & Berliner (1983) specified further conditions for staff development which, if met, would contribute significantly to the value of such activities and the predictibility of their success. The conditions most consistent for projects such as the Coombabah project, described in Part III of this volume, are summarised in Table 9.1 as recommendations for staff development activities for learning and teaching with laptops. Obviously, these recommendations have more general applicability for teachers who teach classes where students have ready access to computers, even though the student-computer ratio may not be 1:1.

EFFECTIVE STAFF DEVELOPMENT

Applying the conditions suggested in Table 9.1 to staff development in computing and computer use in the classroom, one would define an effective staff development program as one which is designed to enhance the teachers' knowledge and skills in ways which lead to changes in their thinking (i.e. planning and decision making) and teaching with computers. These changes in thought and teaching must find support at the school and regional level, and by teachers and educational administrators alike.

Professional development programs should have clearly stated goals (5) which are consistent with the teachers' perceived needs, plans for their work, and classroom teaching conditions (1). The activity should permit variation in the ways teachers participate and apply what they have learnt in their own classrooms (2). The content of instruction in computer use should be concrete (7), and its application to the classroom should be demonstrated by an instructor who is competent in teaching adults and who is able to model using computers in the context of the ongoing curriculum (6). The duration of the program should permit teachers sufficient time to learn, practise, master and apply in their own classrooms the skills and knowledge imparted (8). The professional development program should provide systematic personal guidance and support during the course of training as well as during the period of initial classroom implementation (4). Finally, teachers should receive positive incentives for their participation during the training, implementation and institutionalisation phases (3).

Organisational and Content Features of Professional Development

While the framework provided by Fenstermacher & Berliner (1983) identified important organisational structures and processes for professional development activities, it was not intended to identify the content of staff development for personal computer-based teaching or educational computing specifically. The

Table 9.1 Conditions for Staff Development in Personal Computer Use in Schools[1]

	Condition	Description
(1)	Appropriateness	The activity is consistent with plans teachers have for their work, fits well with classroom conditions, is timely and valued for its usefulness.
(2)	Variability	The activity permits variation in the ways teachers participate and in the ways they use what they have learnt.
(3)	Incentives	The activity provides positive intrinsic incentives to participants, both during the course and during implementation of what has been learnt in the classroom.
(4)	Maintenance	The activity provides systematic and personal support during its duration and during the period of implementation in the classroom.
(5)	Objectives	The activity has clearly stated objectives known both to providers and recipients and clearly related to work demands on the recipients.
(6)	Instructor	The activity is staffed by providers who have competence in teaching adults, and are able to model what they propose teachers should do in their classrooms.
(7)	Application	The content of the activity is sufficiently concrete to make its application to the classroom clear.
(8)	Duration	The activity provides sufficient time for participants to learn, practise, master and apply the content imparted.

literature on the latter is restricted. Few studies have been conducted, and those which have been reported tend to be case studies, rather than comparative studies which systematically varied characteristics of training such as organisation, content, instructional method, support, and incentives. Hence, this chapter will provide some brief suggestions only of different forms of staff development, merely to illustrate some of the range of alternatives discussed in the literature and by Australian practitioners to date.

The most prevalent form of staff development is probably the one-off, short-term workshop for interested teachers. Such workshops are usually offered at a central site and last 2-3 hours per session over a period of a few days. These workshops are often led by computer education experts from a tertiary education institution, sometimes by overseas academics and sometimes by computer

[1] Numbers in parentheses in the text refer to recommendations listed in this Table.

companies. Teachers are taught how to operate the computer and to write elementary programs. Sometimes they are taught about a range of software, less frequently about the selection and/or modification of software and other computer related curriculum materials.

Another approach combines the one-off, short-term workshop with one or a small number of additional, focused workshops which deal with specific topics. A third approach has been to train a small group of teachers who then provide workshops in teaching with computers to their colleagues. These teachers become resource persons, often for particular computer applications, within their school and/or region. They might conduct introductory and more advanced sessions at their school or within the region.

Published case studies and discussions with teachers revealed that, with regard to the content of staff development, teachers are looking for sufficient time to review software and other curriculum materials for use with personal computers, and to plan how to match these materials to the needs of their own students. With respect to organisational factors, time is crucial for most teachers. As noted earlier, staff development in the use of computers requires a time commitment beyond the actual workshops, as teachers personally become familiar with machines and learn to plan instructional applications suitable for their students. The topics on which the majority of novices of educational computing appear to be seeking information are:

1 personal computer operation,
2 computer programming,
3 computer literacy, and
4 selection and evaluation of curriculum materials including software.

Table 9.2 contains major topics which a large proportion of a sample of 1200 teachers requested for staff development activities in a survey conducted by the US National Education Association (1983). These topics are equally important for Australian teachers. The variety of topics highlights the importance of providing alternatives in staff development to meet the needs of individual teachers.

SUMMARY OF IMPORTANT ISSUES AND RECOMMENDATIONS

The topics and organisational features listed in Tables 9.1 and 9.2 could provide a beginning for the formulation of recommendations for the professional development of teachers who use personal computers in their classrooms. Before considering the reactions and recommendations made by the teachers in the project, I would like to focus on some issues and recommendations which I regard as particularly important or which have been debated in the literature.

Table 9.2 Topics and Organisational Features Requested for Staff Development[2]

	Topics
(9)	Operation of personal computer and peripherals
(10)	Computer programming
(11)	Selection and evaluation of curriculum materials
(12)	Modification of materials
(13)	Computer literacy (e.g. history, implications, types of languages)
(14)	Non-instructional uses of the personal computer (e.g. computer-based management)
(15)	Integration of personal computer-based instruction into the curriculum
(16)	Design and authoring of software
(17)	Match of teaching materials with student abilities and learning styles
(18)	Selection of curriculum materials
(19)	Computing curricula and teaching computing
(20)	Development of a user network
(21)	Copyright protection issues
(22)	Instructional uses of personal computers
	Organisational features
(23)	Staff development located at a central site
(24)	Staff development provided in either single or multiple sessions; (25) depending on topics covered
(26)	Instruction provided by outside consultant, teacher or district personnel who meet the instructor condition (6)
(27)	Training adapted to teachers' needs and interests
(28)	Extensive hands-on practice provided

One of the most controversial issues is whether to include computer *programming* in introductory staff development activities, which are aimed at providing teachers with the knowledge and skills needed to use personal computers in instruction. While some advise that programming is to be avoided in the introductory stages of computer training (e.g. Hamolsky, 1983; Nanson, 1982), others assert that programming is the essential component of computer literacy (e.g. Luehrman, 1981). Between these extremes are those who advocate some introduction to a programming language (usually BASIC or Logo) as a way to understand computers and programming (e.g. Page & Wallig, 1983; Widmer & Parker, 1983).

This issue is, obviously, part of a larger concern on the part of educators and others to define computer literacy, which is discussed in more detail in the chapters included in Part II of this volume. Suffice it here to say that the lack of a

2 Numbers in parentheses continue on from Table 9.1 and are recommendations referred to in the text of this chapter.

generally accepted definition of literacy has not prevented interested groups from declaring minimum competencies (e.g. Poirot, 1980; Benderson, 1983).

Perhaps the most frequently addressed organisational issue relating to professional development among American teachers concerns teacher *incentives*. No systematic data are available on this problem with respect to teachers using technology in Australian classrooms. In the USA some school districts use a variety of incentives to maximise teacher participation in staff development activities, publicly available computer courses, conferences and other activities which are designed to broaden their experience and expertise with personal computers. Incentives in these districts include incremental salary credit (e.g. Sheingold, Kane, Endreweit & Billings, 1981; Page & Wallig, 1983), reimbursement for outside courses (e.g. Coburn et al., 1982), release time (National Educational Association, 1983; Office of Technology Assessment, 1982), and new job titles with higher salaries for technically experienced teachers (Office of Technology Assessment, 1982). After initial training, other incentives, such as providing computer resource personnel (Sheingold et al 1981), lending computers to teachers over the weekend and over holidays (Sherman, 1983) and subsidising teachers' purchases of personal computers are provided. While most of the evidence indicates that incentives help motivate teacher participation in staff development and encourage their continued interest in personal computers for instruction, little is known about which incentives or which combination of them are most effective.

TEACHER RECOMMENDATIONS

We asked teachers to describe what they would regard as an ideal staff development program for themselves in relation to the instructional use of personal computers. More specifically, we asked them to comment on the content or topics which should and should not be included, and on organisational features of such staff development, including location, duration and incentives. We also asked the opinion of teachers as to whether pre-service education in educational computing should differ from in-service education, and, if so, in what ways.

Recommendations Regarding Conditions On the whole, Australian teachers regarded the conditions noted in Table 9.1 as essential or highly desirable. The teachers in the SUNRISE classrooms at Coombabah Primary School (n=6) rated these conditions and organisational features, i.e. items (23) to (28) in Table 9.2, of staff development as more important than the actual content. They agreed most strongly in their ratings on conditions and least on content. Individual differences among teachers in their preferences for the actual content of professional development are reported in most studies, as the needs of individuals vary. The only item which all teachers regarded as essential was that the staff development offering should allow for *extensive hands-on experience* (28). The conditions of

appropriateness of the staff development (1) and maintenance, i.e. systematic and personal support during staff development and during the implementation phase (4), were regarded as essential or highly desirable by all teachers. None of them regarded any of the conditions described in items (1) to (8) to be initially unnecessary.

Recommendations for Content The topics most frequently mentioned by teachers were consistent with the findings of the previously noted literature review. Teachers expect professional development activities to focus on the operation of the personal computer, computer programming, selecting and evaluating software, instructional uses of the computer in the classroom, computer literacy, and the integration of the computer with the ongoing curriculum. The teachers in the project were less concerned about administrative uses of the computer.

The teachers in the project regarded programming as an essential component of staff development. A small number, mostly primary school teachers not using Logo, did not want programming included in staff development, and others did not mention programming at all.

The teachers in the SUNRISE classrooms saw the *operation of personal computers and peripherals* (9) and the *integration of personal computer-based instruction into the curriculum* (15) as the two most essential areas for professional development. Next in importance to them was to gain the knowledge which would help them in the *selection and evaluation of courseware* (11) and the skills necessary for the *modification of materials (12).* Of least importance to these teachers were *the knowledge and skills to design and author software* (16), closely followed by *computer literacy* (13), the *development of a user network* and *copyright protection issues* (21). Copyright protection issues are obviously of greatest importance to authors, but this issne is also important for the user of software in the classroom.

The interests and needs of the teachers in the SUNRISE classrooms at Coombabah differed from those of other Australian teachers and those reported in the overseas literature. The SUNRISE teachers were found to have a greater interest in the hardware than in computer literacy and issues of instructional, administrative and personal uses of personal computers. A reason for the former might be that, during the initial stages of the project, teachers had considerable problems with the hardware with little, if any, technical support. The lesser need of these teachers for professional development relating to instructional and management issues might be explained by either a higher level of expertise among them, or their total commitment to Logo and a misperception that reasonable proficiency in this language would obviate the need to reflect on student abilities, learning styles, etc.

Only two of the teachers in the SUNRISE classrooms regarded *computer programming* (10) as an essential topic for staff development. The others rated this area as no more than that *it could be useful.* Two of these teachers suggested

that programming expertise might actually *get in the way* of learning in a Logo environment where students are expected to learn by personally exploring what the computer will enable them to do.

Organisational Recommendations Teachers' preferences were for a series of workshops, held during school hours or after school, located at their school or in close proximity, averaging about 10 hours (the range of suggestions was 1 day to 3 days) in duration with as much hands-on experience as possible. One of the SUNRISE teachers suggested that selected teachers should be given a term of study leave to allow them to attend a more comprehensive course on how to integrate teaching with computers into the curriculum.

The teachers also recommended that professional development activities be offered in a variety of topic areas (so that teachers could attend sessions on only those topics which fulfilled their special needs) and at different levels of sophistication. The teachers in the SUNRISE classrooms were particularly interested in improving their knowledge of Logo, LogoWriter and other word processing packages.

Teachers were unanimous in the view that participation in professional development in educational computing should be voluntary. Apart from a unanimous judgment that *extensive hands-on experience* (28) is essential, and that staff development activities should be held in school time but away from the school, the SUNRISE teachers at Coombabah Primary School expressed fewer strong preferences for specific organisational features than other teachers.

The teachers in the SUNRISE classrooms at Coombabah differed from other teachers who were interviewed, and from the literature, with respect to their view that teachers require very little special training (apart from knowledge of how to use the hardware) in order to start using personal computers in their classrooms. These teachers feel that manuals and books provide sufficient information for teachers and that teachers should be creative enough to invent ways of using different pieces of software in their teaching. One of the most interesting responses received from a teacher at Coombabah was: 'Having the courage to allow children to take Logo and create, *without* the teacher having to be the expert, is the most important part of using Logo.'

Only 5 of over 60 teachers interviewed in Victoria and NSW mentioned *incentives* for the participation in professional development. The teachers who supported incentives felt salary rises to be most appropriate. Half of the teachers in the SUNRISE classrooms at Coombabah felt that incentives would be highly desirable, the other half felt that incentives could be useful. However, all of them reported that professional development is intrinsically motivating for them. The literature shows that in the USA nearly half the teachers surveyed on this matter thought incentives of additional salary or release time to be desirable. Australian and US teachers who oppose incentives for participation in professional development feel that incentives could encourage some teachers to become involved for the wrong reasons.

Pre-service Training All teachers recommended the incorporation of personal computer-based instruction throughout the years of pre-service training of teachers. More than half of the teachers felt that pre-service training programs should differ from in-service. Some recommended more breadth in the pre-service courses, such as learning about different types of computers and computer languages, and exploring a variety of ways in which computers can be used as teaching tools.

The SUNRISE teachers recommended as much exposure as possible *over a period of years*, with an emphasis on word processing (including LogoWriter). Only one of these teachers recommended that in teacher training at colleges and universities computing should be integrated with curriculum areas, and that this might be reinforced by the introduction of a subject such as *computing across the curriculum*.

GENERAL RECOMMENDATIONS FOR PROFESSIONAL DEVELOPMENT

The following recommendations were derived on the basis of the literature on staff development, published case studies and informal surveys of teachers' opinions in Victoria, NSW and Queensland. These recommendations should not be interpreted as prescriptions -- rather, they are presented for the consideration of those who are designing staff development activities which might best meet the needs of teachers and the current constraints of resources.

Recommendations Regarding Organisation[3]

Participation in staff development activities should, whenever feasible, be voluntary.

Initiation of staff development activities should, where possible, be a collaborative effort of teachers and administrators. This would link financial decisions to needs and experiences of the teachers who are implementing the use of computers in classrooms. Teachers collaborating with one another provide added mutual support (4).

The *objectives* of a staff development activity should be clearly stated and understood by both the providers of and participants in staff development activities (5). Ideally, both parties should have input into the definition of objectives. The objectives need to reflect both teachers' needs and regional (state, or national) goals for using computers in schools.

---

3 Numbers in parentheses refer to recommendations in Tables 9.1 and 9.2.

The *appropriateness* condition leads to the recommendation that the staff development activities should meet teachers' needs and plans for their work in a timely manner (1).

The *application* of the content of each staff development activity to the use of computers in the classroom should be clear and concrete (7). This includes provision of software and/or curriculum guides which are immediately applicable to teachers' instructional needs.

The *variability* condition leads to the recommendation that the staff development activity permits teachers to decide whether they will participate, for how long, and how they will apply what they have learnt (2). One way to accomplish this is to individualise instruction as much as possible (27). Another way is to focus each professional development activity on a different topic, and to offer programming as a more advanced course to teachers who wish to acquire this skill. For example, regions might offer courses at different levels, beginning with a core course (9,10,13) and ending with advanced programming. Individualisation of professional development activities, by whatever method, should also help to meet the conditions of appropriateness (1) and application (7).

The *instructor* is, preferably, someone who is or has been a teacher with extensive experience in personal computer-based instruction in the classroom. He/she should be an expert on computers and instructional uses of them, and competent in teaching adults. The instructor should be viewed as competent by participants, but not as *too technical* or out of touch with those whom he or she is teaching.

The *duration* of the staff development activity should be sufficient to permit teachers to learn, practise, master, and apply the skills imparted (3). Although the actual time will vary according to the design of the program, it is important that sufficient time be devoted to introductory activities. The staff development literature suggests that 8 to 10 hours spread over three or four sessions is typical. Although this may be sufficient to show the novice how to operate the machine and review some software, it may fall short of including other important topics, such as programming and how to integrate the computer into instruction in particular curriculum areas.

The *maintenance* condition leads to the recommendation that staff development activities be followed up during the period in which teachers are initially applying the newly acquired skills in their classrooms (4). Many teachers recommend that staff development be ongoing (32), e.g. a multi-session (24) initial workshop with follow-up (30), and allow for plenty of hands-on practice (28). During the phase of initial implementation in the classroom, teachers need a variety of support services and expert resources to assist with hardware repair, evaluation, selection and adaptation of software, as well as with day-to-day troubleshooting. At the very least, teacher networks (ideally, via personal computer and telephone modems) might be established to exchange ideas and experiences concerning personal computer use in the classroom (20). Such

networks are of particular importance during the first year of an innovation and for teachers who are new to implementing computers in their classrooms.

In the US literature, *incentives* are recommended for all phases of staff development (i.e. actual training sessions and follow-up) to support and encourage personal computer use in schools. However, discussions with Australian teachers suggest that the majority of them would not participate in staff development activities just because of incentives. Rather, the major reason for their participation is their high level of interest in and commitment to instructional uses of personal computers. As suggested earlier some incentives might lead to participation for the wrong reasons.

All this is not to suggest that incentives should not be provided. On the contrary, I would suggest that careful consideration be given to the *types of incentives* which might be feasible and appropriate. Release time and salary loadings are the main incentives discussed in the US literature. For many Australian teachers time might be more valuable than money. They may wish to try out many more different ways of using computers than they have time to. Also, as mentioned previously, lending teachers hardware and/or software to take home, or assisting them in the purchase of their own computing equipment through low interest loans or preferential buying arrangements, may be excellent ways of encouraging and supporting the successful implementation of personal computers into Australian classrooms.

Recommendations Regarding Content

Basic professional development activities in educational computing should probably include the following topics: operation of the personal computer (9), selection and evaluation of software and curriculum materials (11), instructional uses of personal computers (22), computer literacy (13), and methods for integrating computers with the ongoing curriculum (15). Such a course might also include computer programming (10), at least to the degree that programming either helps teachers to understand better how the computer operates, or satisfies the variability condition discussed above. In the following sections these recommendations are discussed in detail.

Operation of the personal computer. This would include starting the computer, loading and running programs, keyboard skills, saving work, printing and minor troubleshooting. The time required for a teacher (without prior computing experience) to become a fairly skilled *operator* has been estimated to be in the order of 2 to 6 hours.

Selection and evaluation of software and/or curriculum materials. Teachers need to review a range of materials which are appropriate for the year level of their students and focus on materials which are immediately accessible and available for their use. Material evaluation forms might be developed by schools and regional staff with expert consultation, or evaluation guides might be adopted

which have been developed in other regions/states. (See Chapter 10 for additional details on software and curriculum evaluation.)

Instructional uses of personal computers. Personal computer-based instruction involves more than just instruction which can be delivered by a computer program or disk. Teachers need to acquaint themselves with quite a number of the many uses which can be made of computers as tools, or rather as tool boxes which can be personalised for use, because the computer can be *instructed* to operate in many different ways by the user.

Computer literacy. It is recommended that initial training should aim at a certain level of computer literacy. This includes knowledge about computers and the implications of computing as well as hands-on skills. (This topic is discussed in more detail in Part II of this book.)

Integration of computers with instruction. This involves training in how to integrate what the computer offers with subject content and class activities. Logistics need to be considered, such as rules for student use, transitions between computer and non-computer activities, and classroom arrangements including student grouping. More importantly, teachers may require guidance in how to plan the best utilisation of computers in their teaching. They need sufficient information to begin to make reasonable decisions about matching computing and available software to their instructional goals, the structure of the subject matter, the characteristics of the students, and the content of instruction. Moreover, they need to acquire interactive teaching skills which will help them carry out their plans, monitor and evaluate learning and teaching activities, and make adjustments where required. Of course, these activities are part of what teachers do every day, whether or not they use computers. Computers, however, introduce an additional set of complexities which teachers need to cope with.

Integration involves not only the use of personal computers within the ongoing curriculum: it also involves the adaptation of the curriculum to useful software packages (e.g. Logo) or the adaptation of software to meet important curriculum aims. Lesson plans, introductions to topics, ways of helping students to transfer what they have learnt, and methods for monitoring and evaluating computing activities must be developed.

Computer programming. It is recommended that computer programming be included in introductory professional development to the extent that such knowledge is needed for an understanding of how the computer works, and to understand the basis for applying other skills recommended above, e.g. basic troubleshooting, and computer operation. Thus, some programming will be an essential part of professional development, but perhaps to a lesser extent than many computer users who are not teachers would expect.

The depth to which programming is taught in an introductory professional development course will depend, to a large degree, on the variability condition, i.e. the extent to which particular teachers need to know how to program in order to use the personal computer as an instructional tool. It is likely that mathematics teachers, both in primary and secondary schools, will need more extensive

introductory training in programming than most others, because simple programs can be written as tools for solving mathematical problems. The reason science and social science teachers have not been included here is that it is possible that the more complex data analysis programs or simulations often used in these areas may be too time-consuming for students and teachers to develop as a regular part of classroom computer use. However, these decisions must ultimately be made by the subject teacher for each classroom.

Special Resources

The SUNRISE teachers at Coombabah are asking for easier access to libraries of books related to the use of computers in teaching, and the setting up of a hotline to a local support centre, leading to immediate assistance in hardware and software problems. They recommend that every school should have easy access to at least one person who is expert in instructional computing. This person should be situated at the school or the local school support centre.

All of these teachers commented on the benefits they gained during one term in 1990, when a weekly training session in Logo was conducted for them. They are keen to continue with this type of activity.

Promoting Computer Literacy among Teachers

Subject associations, and state and national teacher associations can serve an important role in promoting computer literacy by making the subject prominent at meetings, in their journals and at conferences.

In most Australian states, support services provide direct assistance to schools and to individual teachers. Services are distributed through regions. In addition to this, local expertise needs to be developed in each school.

Most of the teachers consulted in this study have focused on the the lack of sufficient time allocated to student instruction, for example in mathematics and science. They believe that, even without computers, more time is needed in most subjects areas so that students can explore ideas and experiment. The question arises: Will the introduction of computers exacerbate this problem? The answer is yes, but the extent of the impact cannot be estimated at this stage.

A key ingredient in motivating teachers to use and teach with computers is likely to be by allowing them to see the potential of computing in their own subject areas or primary school classroom, as opposed to just providing them with the machines. There are a number of arguments which might be motivating for teachers. For example, science teachers should probably be convinced that to be a scientist today one must be computer literate.

The argument of job security is also valid. At least a short-term solution to computer literacy in science, mathematics, language and other subject areas

might be not to totally overhaul existing curricula, but to introduce computing as a new component of the subject in its own right, such as computing in mathematics, computing in social studies, etc. Another solution is to provide teachers with revised curriculum guidelines which show them how to use the computer in the attainment of traditional curriculum goals. Computer related activities can broaden the existing curricula rather than be introduced as a new subject. A rewriting of the curriculum for learners who have ready access to computers may well be way down the track.

Frustration among teachers teaching with computers can be minimised by providing more computer resources and meaningful subject and topic related activities. It would be helpful to provide teachers with data on disk with examples of student work collected in other classrooms, case studies showing a range of the work of students of different proficiency, and above all, descriptions of moderation meetings between teachers discussing and validating the assessment of student work.

10

User-friendly Software and Curriculum Materials

The purpose of this chapter is to provide some assistance to teachers in their evaluation of educational computer software and to suggest some ways in which the information and communication gap between educators and the designers of educational computing software might be narrowed. The latter may well be an essential prerequisite for a general improvement in educational computing software.

Teachers, educational administrators and students agree that the ready availability of high quality software and curriculum materials is essential if personal computers are to make a significant contribution to education. Yet the general opinion is that the quality of currently available software is not always good enough, and that its applicability and the extent of curriculum cover can be quite limited. The following factors are amongst those which are likely to have contributed to this state of affairs:

- The development of high quality software is expensive and the extent and the stability of the educational market in Australia is as yet uncertain. The same factors combine to create financial barriers in other countries, including the USA and the UK.
- Design principles currently used to develop educational programs for personal computers tend to be those developed for mainframe machines in the 1960s and early 1970s (cf. Loop & Christensen, 1982). These principles do not necessarily take advantage of the special capabilities of personal computers, nor do they take into consideration the instructional design and research findings in educational theory, cognitive science and related fields which have been produced over the past 20 years.
- The programming languages used (mostly BASIC or Assembly) are not conducive to systematic software design and structured programming methods (e.g. Sommerville, 1982; Wirth, 1973). This makes it more difficult to write long programs and virtually impossible for teachers and other users to modify existing ones.
- Much of the educational software and computer-based curriculum materials are written by people in a type of cottage industry of material development,

where few individuals combine the subject-matter expertise, knowledge of software design principles and the programming skills required to produce non-trivial high quality software (cf. Becker, 1982). Moreover, some software developers have insufficient teaching experience and/or insufficient recent contact with students in actual classrooms to enable them to write *teacher-friendly* software.

- There needs to be more communication and consultation between teachers, the developers of educational software and the designers of curriculum materials for educational computing.

What is easily forgotten is that, as in all production processes, the process of software development in itself can directly influence the quality of the final product.

METHODS OF SOFTWARE DEVELOPMENT

Software developers and educators rarely communicate with one another at the design or development stages of materials. As noted previously, educational software development is largely a cottage industry of one-off efforts produced by individuals in their spare time. A single author, especially when working part-time, is rarely in a position to develop systematic and easily portable sets of tools or utilities for use in multiple programs. Yet such utilities are essential to good quality software, because they include the means of making a program *crash-proof*, the procedures for presenting text or graphics, and the means of handling various types of student and teacher input. Rather than investing the considerable time required to develop such tools, authors often avoid the issue by writing programs which involve only simple student input and which often are not crash-proof.

In this type of development process, content and pedagogy are usually formulated *during* the writing process. The result is that the program might work well enough, but that it might lack coherent overall design. Such programs are difficult to revise and adapt. It is usually easier to write a new program of similar quality from scratch. Only relatively short programs can be rewritten more easily than revised, however. This is why this method of writing programs may be ill-suited for producing software intended to assist the teaching of substantial portions of a subject, or to teach a topic in a sophisticated and adaptive manner. This is particularly true for programs which involve branching and other complex decisions.

An alternative to individual authoring of software is a software design strategy (e.g. Roblyer, 1981; Chambers & Sprecher, 1983) or a production system (e.g. Bork, 1984). This is a multiphase, team process. The development of the software takes place in several phases, including design, development, evaluation and revision. Each of these includes subtasks, such as the specification of learning

objectives, design of learning materials, specification of format, screen layout, coding, and, above all, validation. Each phase involves one or more individuals with expertise appropriate to the demands of the task. The selective use of experts for each task is a key element in design strategies.

Multiple rounds of validation, evaluation and revision are an integral part of design strategies. Such evaluations involve in-house review, as well as testing of the materials with individuals representing the target audience. Revisions and corrections are made on the basis of these formative evaluations. The cycle may include other stages, and may be repeated as often as deemed advisable.

Instructional software utilities should include capabilities for presenting text and graphics on the screen, allowing for a variety of inputs by the student, making programs crash-proof by disabling keys that are inappropriate for a given input, facilitating input analysis, and allowing the storage of user input for evaluation purposes. Clearly, this is an expensive task. However, if done properly, the utilities can be useful for developing all subsequent software and also for simplifying the transport of software from one computer to another. The cost of utilities will then be a once-only investment.

Initially, software designed and created by a team will be more expensive than individually produced software. It is suggested that these costs must be borne, however, since the amount, scope and sophistication of software that will be needed in the coming years can only be created in a manner that: allows the *development of large, but easily revisable programs*, includes *quality control* (i.e. evaluation) facilities, and meets the *instructional requirements* set by curricula and teachers.

EVALUATION OF SOFTWARE

High quality software is examined by its producers to ensure that it is free from factual, linguistic and programming errors. Teachers and curriculum experts must also evaluate it, to ensure that the software fits instructional goals, and is appropriate for the intended audience (i.e. year level, subject area, student characteristics). Teachers will evaluate programs from a number of different perspectives: Firstly, the teacher might quite generally answer questions such as:

- Is the material suitable for the curriculum goals I have in mind for my class?
- Does the program fit the learning approach of the class?
- Could the experience gained by using the computer have been acquired equally well by other means?

Secondly, the programs should be evaluated with several types of different students in mind, including students who tend to follow directions and answer questions with ease, those who tend to have difficulties with directions and those

who tend to read only part of the direction or question. The teacher must carefully observe selected students' interaction with the program.

Thirdly, technical aspects and the friendliness of the program are assessed by examining the clarity and detail of instructions and the ease with which the software can be used without direct teacher supervision. Quality and accessibility of the manual, quality of output and the nature of feedback also need to be considered. Finally, the teacher will seek out published reviews of the software and, where possible, discuss the software with colleagues who have used it with their students. Published reviews and evaluation guides for software are particularly useful. They help the potential user to form a systematic impression of the material and often help clarify for oneself what one actually expects of the software. They also provide the novice with an introduction to the language commonly used in the discussion of the characteristics of computer software.

Teachers who are investigating the potential usefulness and qualities of computer software options certainly need to have information on each of the following:

1 *General characteristics* of the software, including the information it contains, the structure of the information, special features of the program, language requirements and other pre-requisite knowledge required by the user. Questions to be answered include:

   Is the material organised in a reasonable way?
   Does the program contain idiosyncratic symbols?
   Is it easy to use?
   Is the language appropriate for my class?
   Does the software provide learners with choices of path within the learning module?
   Does it contain an effective help sequence?
   Does the material contain modules that vary in difficulty?

2 *Technical characteristics* of the software, including the basic internal workings of the program which might influence student learning in some way. The clarity of instructions, the quality of output and the nature of feedback are examples of the technical characteristics of a program. Hardware requirements and user-friendliness also come under this heading. Questions to be answered include:

   Does the program provide informative feedback?
   Is it difficult to crash?
   Can it be restarted easily when it stops?

3 *Physical characteristics* of the software. This might include screen layout. Screens should not be cluttered with crowded text or graphics. Rather, the material on the screen should be arranged in a well spaced manner and thus facilitate the student's visual scanning of the information.

   Does the program utilise appropriate *frame* size, i.e. neither too large nor too small?
   Is it boring to use?
   Is the software accompanied by comprehensible documentation in the form of manuals?

4 *Educational aspects* of the software. These relate to what teachers consider in every educational process, i.e. objectives, contents, activities, and evaluation. Other educational aspects include the compatibility of the program content with other materials used by the teacher in a particular learning area, and the degree to which the software actively engages the student, encourages creativity and motivates the student. The speed with which the program reacts and the sequencing of the program components also influence the educational process. Other areas of importance here include:

   Does the software hold the learner's attention?
   Does it keep the student active?
   Does it use graphics in a pedagogically sound way?
   Does it provide a means of keeping track of the students' progress?

Access to Software for Review

Systematic and efficient means are needed to provide teachers with information about software, and developers with feedback about the users' reactions to their materials. Ideally, teachers should have direct access to the software, in order to review it themselves. Where this is not possible, extensive and objective reviews by other teachers should be available. This does not deny the fact that software publishers must have some safeguard that their copyright products will not be used without authorisation.

Software libraries, which are readily accessible to teachers (perhaps *on-line*) and contain programs with evaluations, can solve these problems to a large extent. They could be an excellent resource for teachers. At the same time they offer a means of evaluating software without violating copyright restrictions.

Software attributes which teachers expect generally include the following:

- general *user-friendliness*, such as clarity of instructions, ease of operation,
- appropriateness for the teacher's purposes, including modifiability by the teacher,
- adaptability to curriculum content,
- the program makes appropriate demands on the students,

- the software fully uses the capabilities of the hardware.

Friendliness. An essential attribute of a satisfactory piece of instructional software is that it will not stop (*crash*) as a result of inappropriate input. Educational software needs to be *crash-proof* and sufficiently *bug-free* to allow normal operation. Another friendliness attribute relates to the teacher's option to modify the software to fit more closely with the curriculum or the needs of individual students. This capability concerns the possibility of customising software for specific purposes and situations. The teacher may wish to vary the number or type of mathematics problems, specific information given in the program, or the length of a session. Other modification capabilities include the skipping of certain instructions, personalising an interaction and networking. Most teachers would also require some sort of record-keeping facility in the software.

THE ROLE OF TEACHERS IN SOFTWARE DESIGN

Ideally, teachers should have considerable input into the design of educational software. Teachers have expert knowledge with respect to the prerequisite skills which are required to master subject content. Therefore they, rather than a programmer, should determine and communicate the prerequisite knowledge and skills required for the use of a program. Including teachers in the software design team will assure

- that the program content is non-trivial and free from content errors,
- that the level, pace and mode of computer-based instruction are appropriate for the intended audience, and
- that the program actively involves the student in a learning process.

Involving teachers in software design also makes it more likely that critical thinking and problem solving skills will be emphasised in the program. Teachers can help assure that programs have diagnostic and feedback capabilities. Because teachers are familiar with the curriculum and with popular textbooks, they can, as members of the design team, coordinate software with other materials.

Because they are most familiar with learning demands in particular subject areas, teachers will make the best consultants on how other teachers may wish to modify a particular piece of software, or customise a database. Finally, teachers may provide the designers of educational software with useful judgments as to whether a program fulfils the pedagogical goals set for it before the program is evaluated empirically.

The lack of good software is frequently cited by some teachers as a reason for their reluctance to incorporate computer-based learning into their own classrooms. One reason for this is the difficulty that teachers face in obtaining

potentially useful software for review. Many publishers of software, unlike book publishers, are quite reluctant to send out review copies of their products, and only a few program producers have resolved the problem by giving out small exemplar programs that allow teachers to understand what the complete program can offer.

EVALUATION CRITERIA FOR COMPUTER SOFTWARE AND RELATED MATERIALS

As noted above, some sort of evaluation form or checklist is an excellent idea, because it leads to a more systematic assessment of the materials and might actually help teachers to clarify for themselves what exactly they are looking for in instructional software. The NSW Department of Education's Computer Education Unit (1985) produced a useful software evaluation form, which consists of a cover sheet and an evaluation checklist. The cover sheet forms a summary review of the software package. The evaluation checklist, reproduced at the end of this chapter, leads to a report on the material under four main sections, namely content, classroom application, program features and support materials.

1 *Content* describes subject matter, aims and objectives, bias, concepts introduced, relevance, flexibility and required teaching style.
2 *Classroom application* describes preparation required by the students and teacher, prerequisite knowledge, operation in the classroom, technical details and follow-up activities.
3 *Program features*. This section shows how the software operates, and includes data input procedures, presentation of material, ways in which the software can be modified, and a report on the structure of the software.
4 *Support materials*. Under this heading technical and operating manuals and other materials intended for teacher and student are discussed with special reference to their availability and quality.

A Guide

The following guide was produced by the Computer Education Unit in the NSW Department of Education, Division of Services, in 1985 and provides a useful example. The NSW Department of Education has declared this material *free for copying for educational purposes if source is acknowledged*.

This evaluation form consists of a cover sheet and a 6-page checklist. All relevant summary comments from sections of the checklist are transferred to the cover sheet. The cover sheet is made up of two pages. The first page provides space for the listing of factual information concerning the software, e.g. title, author, type of computer, price, etc. The second page contains a summary of the

evaluator's opinion of the package. The checklist allows for detailed comments on the software from four points of view:

1 Content: i.e. Aims and objectives, subject-matter concepts introduced, bias, relevance, flexibility and teaching style.

2 Classroom application: i.e. Preparation required by the students and teachers, prerequisite knowledge, operation in the classroom, technical details and follow-up activities.

3 Program features: i.e. Operation of the software, data input, presentation of material, modifiability of the software and structure of the programs.

4 Support materials: Technical and operating manuals, availability and quality of materials and packaging.

The following procedure is suggested for use of the cover sheet and the checklist:

Step 1: Skim the documentation which comes with the software and note important points.

Step 2: Complete the first page of the cover sheet as far as you can. Most of the required information will be found in the documentation and package itself.

Step 3: Work through the package to obtain an overall feel for its structure and operation. Behave as a successful user might, avoiding intentional errors. If, at this stage, you gain the impression that a detailed evaluation of the package is not warranted or necessary, complete the cover sheet. If the package is to be evaluated in detail go to Step 4.

Step 4: Work through the program, making deliberate mistakes to test error-handling capabilities and relevance of feedback. Mistakes might include incorrect responses to questions, typing errors, or failure to follow directions.

Complete the checklist. Note that in most instances it will not be possible to complete the different parts of the checklist in sequence.

Step 5: Summarise the evaluation made on the basis of the checklist on page 2 of the cover sheet.

N.S.W. Department of Education
Division of Services
Computer Education Unit

SOFTWARE EVALUATION

COVER SHEET

Computer(s) _____

Curriculum Area(s) _____

Topic Area(s) _____

Ability Group Age Group _____

Title _____

Author Publisher _____

Price(s) $ ISBN: _____

Date/Version Dewey _____

Related Packages _____

Supplier(s) _____

Address _____

_____ Phone _____

Copyright	Yes [ ]	OR Public Domain [ ]
Permission to make back-ups	[ ]	
Back-ups available	[ ]	Cost $_____
Site licence available	[ ]	Cost $_____
Network version available	[ ]	Cost $_____
Permission to make multiple copies	[ ]	Cost $_____
Demonstration version available	[ ]	
Available on approval	[ ]	
Permission to copy manuals	[ ]	
Permission to copy worksheets	[ ]	

Media: Cassette [ ] Disk [ ] ROM chip [ ] Cartridge [ ]

Required Hardware: Computer _____ Memory _____ K
Cassette [ ] Disk [ ] Printer [ ] Colour Monitor [ ]

Other (Describe) _____

Optional Hardware: Printer [ ] Disk [ ] Colour Monitor [ ]

Other (Describe) _____

Level of Evaluation: 1- Screened
2- Full evaluation by subject specialist
3- Field tested

Evaluated by (full name) _____ Date _____

Contact point (school or phone) _____

THIS EVALUATION DOES NOT NECESSARILY REFLECT THE VIEWS OF THE COMPUTER EDUCATION UNIT OR THE DEPARTMENT OF EDUCATION

PAGE 1

Type of Software

Blackboard	[ ]	Simulation	[ ]	Database Management	[ ]
Drill	[ ]	Problem Solving	[ ]	Word Processor	[ ]
Diagnostic	[ ]	Investigation	[ ]	Spreadsheet	[ ]
Tutorial	[ ]	Prepared Database	[ ]	Programming Language	[ ]
Game	[ ]			Other Utility	[ ]

Description _____

_____

_____

Educational Objectives _____

_____

Method of Use

Individual	[ ]	Introduction to Topic	[ ]	Reinforcement	[ ]
Small Group	[ ]	Concept Development	[ ]	Remediation	[ ]
Class	[ ]	Review	[ ]	Extension	[ ]
Teacher	[ ]	Assessment	[ ]	Enrichment	[ ]

Estimated Student Time To Complete _____

EVALUATION SUMMARY

[I] Impressive [S] Satisfactory [U] Unsatisfactory

Content Rating [I S U]

_____

Program Features Rating [I S U]

_____

Classroom Application Rating [I S U]

_____

Support Materials Rating [I S U]

_____

_____

RECOMMENDATION

Excellent Package	– Recommend without hesitation	[ ]
Good Package	– Consider purchase	[ ]
Fair	– But might want to wait for better	[ ]
Not useful	– Not recommended	[ ]

Overall Comments _____

_____

_____

EVALUATION CHECKLIST

CONTENT

	[✓] YES	[X] NO	[O] NOT APPLICABLE COMMENT
OBJECTIVES			
Is the purpose of the package stated?	[ ]		_____
Are the objectives clearly stated?	[ ]		_____
Are these objectives educationally worthwhile?	[ ]		_____
Are these objectives feasible?	[ ]		_____
Is there a means of assessment of objectives?	[ ]		_____
CONTENT			
Is the content accurate?	[ ]		_____
Is there educational value in the content?	[ ]		_____
Is the content level suitable for the target audience?	[ ]		_____
Are student prerequisites adequately described?	[ ]		_____
Does the program hold the students' attention?	[ ]		_____
Is the reading level appropriate?	[ ]		_____
Are grammar, spelling and expression acceptable?	[ ]		_____
Are special symbols or terminology used appropriately?	[ ]		_____
APPLICABILITY/ADAPTABILITY			
Is the content applicable to the N.S.W. curriculum?	[ ]		_____
Is the content adaptable to a range of curriculum areas?	[ ]		_____
Does the program relate to other learning activities?	[ ]		_____

COMMENTS: _____

_____

_____

_____

_____

_____

CONTENT

[√] YES [X] NO [O] NOT APPLICABLE

TEACHING STRATEGY COMMENT

What is the teaching style? (e.g. Inductive/Deductive, Didactic/Heuristic, Divergent/Convergent, Pupil/Teacher Centred) _____

_____

_____

_____

_____

Is the style appropriate to the objectives? [ ] _____

BIAS

Does the program present positive images and role models for all students? [ ] _____

Is the diversity within a group reflected? [ ] _____

Does the program present a non-violent point of view? [ ] _____

Is the language appropriate and non-disparaging? [ ] _____

Which groups of people are omitted or ignored? _____

Which people are shown as having stereotyped characteristics and roles? _____

Which people are seen as being active/passive? _____

Which people are seen as being dominant/subservient? _____

Which people are seen as being superior/inferior? _____

What values and attitudes are being rewarded or reinforced? _____

Is the inclusion of groups representative or trivial? [ ] _____

Are the role models portrayed trivialised or valued and respected? [ ] _____

COMMENTS: _____

_____

_____

_____

_____

PAGE 4 TRANSFER ALL RELEVANT POINTS TO COVER SHEET

CLASSROOM APPLICATION

PREPARATION COMMENT

What preparation is required by the teacher? _____

_____

What preparation is required by the student? _____

_____

Are set-up procedures adequately described? [ ] _____

Is a demonstration program supplied? [ ] _____

Is a sample of expected output supplied? [ ] _____

OPERATION

What is the role of the teacher? _____

_____

Is the program flexible in application? [ ] _____

Are suitable non-computer activities suggested? [ ] _____

Do students use other materials during the session? [ ] _____

Can students achieve objectives in time available? [ ] _____

Is there the ability to quit and resume? [ ] _____

Are records kept for student and/or teacher? [ ] _____

Are management functions shielded from the user? [ ] _____

Are teachers' files secure? [ ] _____

Are there instructions for the student to end the session? [ ] _____

TECHNICAL

Can sufficient records be accommodated? [ ] _____

Can separate disks be used for data? [ ] _____

Can the disk be removed from the drive after initial loading? [ ] _____

FOLLOW-UP

Are suitable follow-up activities suggested for students? [ ] _____

Are suitable follow-up activities suggested for the teacher? [ ] _____

COMMENTS: _____

_____

TRANSFER ALL RELEVANT POINTS TO COVER SHEET PAGE 5

PROGRAM FEATURES

	[✓] YES	[X] NO	[O] NOT APPLICABLE COMMENT
INPUT			
Is the method for data input consistent?	[ ]		
Are there safeguards against all input errors?	[ ]		
Is on-screen help available?	[ ]		
Are adequate input prompts given?	[ ]		
Are hints available when needed?	[ ]		
Is keyboard response fast enough?	[ ]		
Is the complexity of input appropriate?	[ ]		
Is the structure of input natural?	[ ]		
Can input errors be corrected easily and immediately?	[ ]		
Are special keys used consistently throughout the program?	[ ]		
Is a wide range of appropriate responses accepted?	[ ]		
Is there adequate feedback?	[ ]		
Is feeedback personalised?	[ ]		
Does the program use any special input devices?	[ ]		
PRESENTATION OF MATERIAL			
Is screen layout consistent throughout?	[ ]		
Is the screen layout free of distractions?	[ ]		
Is the screen presentation clear and uncluttered?	[ ]		
Is the amount of text presented appropriate?	[ ]		
Does the presentation of material avoid unnecessary repetition?	[ ]		
Are colour, graphics, animation and sound used effectively?	[ ]		
Is it free of audio or other distractions?	[ ]		
Are lengthy delays flagged by a message?	[ ]		
Is the rate of presentation of material under user control?	[ ]		
Is use made of output devices other than the screen?	[ ]		

COMMENTS: _____

PAGE 6 TRANSFER ALL RELEVANT POINTS TO COVER SHEET

PROGRAM FEATURES

	[✓] YES	[X] NO	[O] NOT APPLICABLE COMMENT
REINFORCEMENT			
Is there positive reinforcement?	[ ]		
Is reinforcement varied in form and content?	[ ]		
Are rewards appropriate?	[ ]		
Are appropriate responses only reinforced?	[ ]		
Does the program involve the student sufficiently?	[ ]		
Does the program have a high motivational value?	[ ]		
OPERATION			
Is the method of operation of the program consistent?	[ ]		
Is the method of operation easy to learn?	[ ]		
Is the operation of the program simple enough?	[ ]		
Is the operation of the program straightforward and varied enough to prevent tedium?	[ ]		
Is the speed of operation of the program appropriate?	[ ]		
STRUCTURE			
Is the structure of the program understandable by users?	[ ]		
Is the method of control of movement through the program consistent?	[ ]		
Can the sequence be controlled by the user?	[ ]		
Can the sequence be controlled by the user?	[ ]		
Are students able to quit program sections?	[ ]		
Can the program be stepped forward and backward?	[ ]		
Are instructional steps of appropriate size?	[ ]		
Is the presentation of material under teacher control?	[ ]		
Can the content be altered by the teacher?	[ ]		
Can the sequence and/or content of the program be different each time it is used?	[ ]		

COMMENT _____

TRANSFER ALL RELEVANT POINTS TO COVER SHEET PAGE 7

SUPPORT MATERIALS

[✓] YES [X] NO [O] NOT APPLICABLE

OPERATION	COMMENT
Is there an operating instructions manual?	
Is the manual sufficiently comprehensive?	
Is the manual logically arranged and easily understood?	
Are copying details given if appropriate?	
Is a sample program run given?	
CURRICULUM	
Is there a teacher's manual with suggested teaching strategies?[	
Are there student workbooks or worksheets?	
Does the documentation (language and illustrations) show a balanced view of groups in society?	
Are reading and research materials supplied?	
Are there references to N.S.W. curriculum materials?	
TECHNICAL	
Is the documentation accurate?	
Is there a description of the structure of the program?	
Are any other useful technical details supplied?	
PACKAGING	
Is the documentation satisfactorily bound?	
Does the packaging adequately protect the disk or cassette? [	
Is the packaging suitable for storage in the library?	

COMMENTS: _____

PAGE 8 TRANSFER ALL RELEVANT POINTS TO COVER SHEET

References

Abelson, H. & diSessa, A. (1981). *Turtle geometry: The computer as medium for exploring mathematics*. Cambridge, MA: MIT Press.

Ainley, J. & Goldstein, R. (1988). *Making Logo work: A guide for teachers*. Oxford: Basil Blackwell.

Anderson, J.R. & Reiser, B. (1985). The LISP tutor. *Byte*, **10**, 159-175.

Anderson, R.C. (1984). Role of reader's schema in comprehension, learning and memory. In R.C. Anderson, J. Osborn & R.J. Tierney (Eds), *Learning to read in American schools: Basal readers and content texts*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Anderson, R., Klassen, D., Krohn, K. & Smith-Cunnien, P. (1983). *Computer awareness and literacy: An empirical assessment*. Minneapolis, MN: Educational Computing Consortium, 1983.

Anderson, S., Ball, S., Murphy, R. & Associates (1975). *Encyclopaedia of Educational Evaluation*. San Francisco: Jossey-Bass.

Apple, M.W. (1982). Curricula form and the logic of technical control: Building the possessive individual. In M.W. Apple (Ed.), *Cultural and economic reproduction in education*. London: Routledge & Kegan Paul.

Arnold, S. (1991). Anyone for MILO? *The Australian Mathematics Teacher*, **47**, 14-17.

Atwood, M.E. & Ramsey, H.R. (1978). *Cognitive structures in the comprehension and memory of computer programs: An investigation of computer debugging*. Technical Report TR-78-A210. Alexandria, VA: U.S. Army Research Institute for the Behavioral Sciences.

Australian Council for Educational Research (1991). *ACER Co-operative Entry Program 1991*. Hawthorn, Victoria: Australian Council for Educational Research.

Barr, R. & Dreben, R. (1977). Instruction in classrooms. in L.S. Shulman (Ed.), *Review of Research in Education*, **5**, 89-162.

Barr, A. & Feigenbaum, E.A. (1982). *Handbook of artificial intelligence*. Los Altos, CA: William Kaufman.

Becker, H.J. (1982). Microcomputers in the classroom: Dreams and realities. Report No. 319. Baltimore, MD: Johns Hopkins University Center for Social Organization of Schools.

Becker, H.J. (1983-84). *School uses of microcomputers*. Reports from a national survey. Baltimore, MD: Johns Hopkins University.

Becker, H.J. (1987). The importance of a methodology that maximizes falsifiability: Its applicability to research about Logo. *Educational Researcher*, **16**, 11-16.

Benderson, A. (1983). *Computer literacy*. Princeton, N.J.: Educational Testing Service.

Berger, P. & Luckman, T. (1967). *The social construction of reality*. Harmondsworth: Penguin.

Berman, P. & McLaughlin, M.W. (1978). *Federal programs supporting educational change. Volume 8: Implementing and sustaining innovations*. Report R-1589/8-HEW. Santa Monica, CA:The Rand Corporation.

Billings, K. (1983). Research on school computing. In M.T. Grady & J.D. Gawronsky (Eds), *Computers in curriculum and instruction*. Alexandria, VA: Association for Supervision and Curriculum Development.

Black, Sir Herman (1988). *The teaching of critical thinking*. Opening address at the NSW Institute for Educational Reasearch's Conference on Critical Thinking held on 27 May 1988 at the University of Sydney.

Bork, A. (1984). Computers and the future: Education. *Computers in Education*, **8**, 1-4.

Borko, H., Shavelson, R.J. & Stern, P. (1981). Teachers' decisions in the planning of reading instruction. *Reading Research Quarterly*, **16**, 449-466.

Botkin, J.W., Elmandjra, M. & Malitza, M. (1979). *No limits to learning*. New York: Pergamon Press.

Boyle, C.F. & Anderson, J.R. (1984). *Acquisition and automated instruction of geometry proof skills*. Paper presented at the annual meeting of the American Educational Research Association, New Orleans, LA.

Brooks, R.E. (1980). Studying programmer behavior experimentally: The problems of proper methodology. *Communications of the ACM*, **23**, 207-213.

Brown, A.L. (1985). Mental orthopaedics, the training of cognitive skills: An interview with Alfred Binet. In S.F. Chipman, J.W. Segal & R. Glaser (Eds), *Thinking and learning skills: Research and open questions*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Brown, A.L., Bransford, J., Ferrara, R & Campione, J. (1983). Learning, remembering and understanding. In P.H. Musson (Ed.), *Handbook of child psychology*. 4th edition. New York: Wiley.

Brown, A.L., Campione, J.C. & Day, J.D. (1981). Learning to learn: On training students to learn from texts. *Educational Researcher*, **10**, 14-21.

Brown, J.S. (1984a). *Idea amplifiers: New kinds of electronic learning environments*. Palo Alto, CA: Xerox Palo Alto Research Center, Intelligent Systems Laboratory.

Brown, J.S. (1984b). Process versus product: A perspective on tools for communal and informal electronics learning. In S. Newman & E. Poor (Eds), *Report from the learning lab: Education in the electronic age*. New York: Educational Broadcasting Corporation, WNET.

Brown, J.S., Burton, R. & deKleer, J. (1982). Pedagogical natural language, and knowledge engineering techniques in SOPHIE I, II and III. In D. Sleeman & J.S. Brown (Eds), *Intelligent tutoring systems*. New York: Academic Press.

Bruner, J.S. (1987). *Actual minds, possible words*. Cambridge: Cambridge University Press.

Bruner, J.S. (1983). *Child's talk: Learning to use language*. Oxford: Oxford University Press.

Bruner, J.S. (1964a). Some theorems on instruction illustrated with reference to mathematics. In the 63rd Yearbook of the National Society for Studies in Education, *Theories of learning and instruction*. Chicago: University of Chicago Press.

Bruner, J.S. (1964b). The course of cognitive growth. *American Psychologist*, **19**, 1-15.

Bruner, J.S. (1966). *Towards a theory of instruction*. New York: W.W. Norton & Co.

Bruner, J.S. & Haste, H. (1987). *Making sense: The child's construction of the world*. London: Methuen.

Bruner, J.S. & Tajfel, H. (1961). Cognitive risk and environmental change. *Journal of Abnormal and Social Psychology*, **62**, 231-241.

Brush, L.E. (1980). *Encouraging girls in mathematics*. Cambridge, MA.: Abt Books.

Burns, G., Cook, M. & Dubitsky, B. (1982). The Logo project at Bank Street: A perspective on children, teachers and computers. Paper presented at the National Educational Computing Conference, Kansas City, MO.

Burrowes, Y.S. (1985). *Logo in high school: Successes, failures and future concerns. Logo 85 Pre-Proceedings*. July (quoted in Harper, D. (1989). *Logo: Theory and practice*. Pacific Grove, CA: Brooks Cole Publishing Company.)

Burstein, L. (1980). The analysis of multilevel data in educational research and evaluation. *Review of Research in Education*, **8**, 158-236.

Burton, G. (1979). Regardless of sex. *The Mathematics Teacher*, **72**, 261-270.

Burton, R.R. (1981). DEBUGGY: Diagnosis of errors in basic mathematical skills. In D.H. Sleeman & J.S. Brown (Eds), *Intelligent tutoring systems*. New York: Academic Press.

Chambers, J.A. & Sprecher, J.W. (1983). Computer assisted instruction: Current trends and critical issues. *Communications of the ACM*, **23**, 332-342.

Chambers, S.M. & Clarke, V.A. (1987). Is inequity cumulative? The relationship between disadvantaged group membership and students' computing experience, knowledge, attitudes and intentions. *Journal of Educational Computing Research*, **3**, 495-518.

Champagne, A.B. & Chaiklin, S. (1985). Interdisciplinary research in science learning: Great prospects and dim future. Paper presented at the annual meeting of the American Educational Research Association, Chicago, IL.

Chandler, D. (1984). *Young learners and the microcomputer*. Milton Keynes: Open University Press.

Char, C., Freeman, C. & Hawkins, J. (1985). Incorporating database software into the classroom context: An ethnographic study. Paper presented at the

annual meeting of the American Educational Research Association, Chicago, IL.

Char, C., Hawkins, J., Wootten, J., Sheingold, K. & Roberts, T. (1983). *The vogue of Mimi: Classroom case studies of software, video and print materials.* Report to the U.S. Department of Education. New York: Bank Street College of Education, Center for Children and Technology.

Chi, M.T.H., Feltovich, P.J. & Glaser, R. (1980). Categorization and representation of physics problems by experts and novices. *Cognitive Science*, **5**, 121-152.

Chung, J. (1991). Collaborative learning strategies: The design of instructional environments for the emerging new school. *Educational Technology*, **21**, 15-22.

Clarke, V. (1989). Involving girls in programming. *Australian Educational Computing*, 4, 21-24.

Clement, C. (1984). Analogical reasoning and learning computer programming. Paper presented at the Harvard University Conference on Thinking, Cambridge, MA.

Clements, D.H. & Gullo, D.F. (1984). Effects of computer programming on young children's cognition. *Journal of Educational Psychology*, **76**, 1051-1058.

Coburn, P., Kelman, P., Roberts, N., Snyder, T.F.F., Watt, D.H. & Weisner, C. (1982). *Practical guide to computers in education*. Reading, MA: Addison-Wesley.

Cohen, P.R. & Feigenbaum, E.A. (1982). *Handbook of artificial intelligence*. Vol. 3. Los Altos, CA: W. Kaufmann.

Cole, M. & Griffin, P. (1980). Cultural amplifiers reconsidered. In D.R. Olson (Ed.), *The social foundations of language and thought: Essays in honor of Jerome S. Bruner*. New York: W.W. Norton.

Commonwealth Schools Commission (1984). Teacjomp. Learning and computers: 1984 information kit. Canberra: Commonwealth Schools Commission.

Conference Board of the Mathematical Sciences (1983). *The mathematical sciences curriculum K-12: What is still fundamental and what is not.* Washington, DC: Conference Board of the Mathematical Sciences (USA).

Cook-Gumperz, J. (1986). *The social construction of literacy*. New York: Cambridge University Press.

Corno, L. & Mandinach, E.B. (1978). Using existing classroom data to explore relationships in a theoretical model of academic motivation. *Journal of Educational Research*, **77**, 141-153.

Crawford, K.P. (1988). New contexts for learning mathematics. In A. Borbas (Ed.), *Proceedings of the Twelfth Annual Conference of the International Group for Psychology of Mathematics Education*. Budapest, Hungary: 239-246.

Culley, L. (1988). Girls, boys and computers. *Educational Studies*, **14**, 3-8.

Cyert, R.M. & Mowery, D.C. (Eds). (1987). *Technology and employment*. Washington, DC: National Academy Press.

Daiute, C. (1985). *Writing and computers*. Reading, MA.: Addison-Wesley.

Dalbey, J. & Linn, M.C. (1985). The demands and requirements of computer programming: A literature review. *Journal of Educational Computing Research*, **1**, 253-274.

Dalbey, J. & Linn, M.C. (1986). Cognitive consequences of programming: Augmentations to basic instruction. *Journal of Educational Computing Research*, **2**, 75-93.

Dalbey, J., Tourniaire, F. & Linn, M.C. (1986). Making programming instruction cognitively demanding: An intervention study. *Journal of Research in Science Teaching*, **23**, 427-436.

Dansereau, D.F. (1985). Learning strategies research. In J.W. Segal, S.T. Chipman & R. Glaser, *Thinking and learning skills*. Vol. 1: Relating instruction to research. Hillsdale, NJ: Lawrence Erlbaum Associates.

Davis, R. & Lenat, D.B. (1981). *Knowledge-based systems in artificial intelligence*. New York: McGraw-Hill.

Deboer, G.E. (1984) A study of gender effects in the science and mathematics course taking behavior of a group of students who graduated from college in the late 1970s. *Journal of Research in Science Teaching*, **21**, 95-103.

de Bono, E. (1985). The Cort thinking program. In J.W. Segal, S.F. Chipman & R. Glaser (Eds), *Thinking and learning skills*. Vol. 1: Relating instruction to research. Hillsdale, NJ: Lawrence Erlbaum Associates.

Dede, C. (1986). Computers in schools: Educational and social implications. In T. Forester (Ed.), *The information technology revolution*. Oxford: Basil Blackwell.

Dewey, J. (1902). *The child and the curriculum*. Chicago: University of Chicago Press.

Dewey, J. (1915). *The school and society*. Chicago: University of Chicago Press.

Dewey, J. (1938). *Experience and education*. New York: Collier Books.

Dillon, J.T. (1984). The character of inquiry in curriculum. Paper presented at the Aunual Meeting of the American Research Association, New Orleans, April.

Dilthey, W. (1976). The construction of the historical world in the human studies. In H.P. Rickman (Ed.), *W. Dilthey, selected writings*. New York: Cambridge University Press.

diSessa, A.A. (1982). Unlearning Aristotelian physics: A study of knowledge-based learning. *Cognitive Science*, **6**, 37-75.

diSessa, A.A. (1983). Phenomenology and the evolution of intuition. In D. Gentner & A. Stevens (Eds), *Mental models*. Hillsdale, NJ: Lawrence Erlbaum Associates.

diSessa, A.A. (1986). Artificial worlds and real experience. *Instructional Science*, **14**, 207-227.

diSessa, A.A. (1988a). What will it mean to be educated in 2020? In R.S. Nickerson & P.P. Zodhiates (Eds). *Technology in education: Looking toward 2020.* Hillsdale, NJ: Lawrence Erlbaum Associates.

diSessa, A.A. (1988b). Knowledge in pieces. In G. Forman & P.B. Pufall (Eds), *Constructivism in the computer age*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Donaldson, M. (1978). *Children's minds*. Glasgow: Fontana-Collins.

Dunn, Lloyd M. & Dunn, Leota M. (1981). *Peabody Picture Vocabulary Test* (Rev. ed.). Circle Pines, MN: American Guidance Service.

Eakins, B.W. & Eakins, R.G. (1978). *Sex differences in human communication*. Boston: Houghton Mifflin.

Eastman, S.T. & Agostino, D.E. (1986). Commanding the computer: Functions and concepts of videotex for eighth grade students. *Journal of Research and Development in Education*, **19**, 49-57.

Eastman, S.T. & Krendl, K. (1987). Computers and gender: Differential effects of electronic search on students' achievement and attitudes. *Journal of Research and Development in Education*, **20**, 41-48.

Easton, P.A. (1989). Restructuring learning environments: Lessons from the organization of post-literacy programs. *International Review of Education*, **35**, 423-444.

Education Development Center (1988). *The Literacies Institute*. Technical proposal submitted to the Andrew Mellon Foundation, Newton, MA. Washington, DC: Education Development Center.

Edwards, D. (1990). Discourse and development of understanding in the classroom. In B. Barrett & E. Scanlon (Eds). *Computers and learning*. London: Methuen.

Edwards, D. & Mercer, N. (1987). *Common knowledge*. London: Methuen.

Emihovich, C. (1989). Learning throngh sharing: Peer collaboration through Logo instruction. In C. Emihovich (Ed.), *Location learning: Ethnographic perspectives on classroom research*. Norwood, NJ: Ablex.

Emihovich, C. & Miller, G.E. (1988a). The effects of Logo and CAI on black children's achievement, reflectivity and self-esteem. *The Elementary School Journal*, **88**, 473-487.

Emihovich, C. & Miller, G.E. (1988b). Learning Logo: The social context of cognition. *Journal of Curriculum Studies*. **20**, 57-70.

Emihovich, C. & Miller, G.E. (1988c). Talking to the turtle: A discourse analysis of Logo instruction. *Discourse Processes*, **11**, 183-201.

Equal Opportunies Commission (1983). *Information technology in schools*. Manchester: Equal Opportunities Commission.

Erickson, G.L. & Erickson, L.J. (1984). Females and science achievement: Evidence, explanations and implications. *Science Education*. **38**. 63-89.

Ericsson, K.A. & Simon, H.A. (1984). *Protocol analysis: Verbal reports as data*. Cambridge, MA: MIT Press.

Feigenbaum, E.A. & McCorduck, P. (1983). *The fifth generation: Artificial intelligence and Japan's computer challenge to the world*. Reading, MA: Addison-Wesley.

Fenstermacher, G.D. & Berliner, D.C. (1983). *A conceptual framework for the analysis of staff development*. Santa Monica, CA: The Rand Corporation.

Feuerstein, R., Jensen, M., Hoffmann, M.B. & Rand, Y. (1985). Instrumental enrichment, an intervention program for structural cognitive modifiability. Theory and practice. In J.W. Segal, S.F. Chipman & R. Glaser (Eds). *Thinking and learning skills*. Vol. 1: Relating instruction to research. Hillsdale, NJ: Lawrence Erlbaum Associates.

Feurzeig, R., Horwitz, P. & Nickerson, R.S. (1981). *Microcomputers in education*. Report No. 4798. Prepared for the Department of Health, Education and Welfare, National Institute of Education, and Ministry for the Development of Human Intelligence, Venezuela. Cambridge, MA: Bolt, Beranek & Newman.

Feurzeig, R., Papert, S., Bloom, M., Grant R. & Solomon, C. (1969). *Programming languages as frameworks for teaching mathematics*. Report No. 1899. Cambridge, MA: Bolt, Beranek & Newman.

Fey, J.T. (1984). *Computing and mathematics; The impact on secondary school curricula*. College Park, MD: National Council of Teachers of Mathematics.

Finlayson, H.M. (1984). The transfer of mathematical problem solving skills from Logo experience. Research Paper No. 238. University of Edinburgh: Department of Artificial Intelligence.

Flavell, J.H. (1977). *Cognitive development*. Englewood Cliffs, NJ: Prentice-Hall.

Flavell, J.H. (1979). Metacognition and cognitive monitoring: A new area of research. *American Psychologist*, **10**. 906-911.

Fox, L.H. (1977). The effects of sex role socialization on mathematics participation and achievement. In L.H. Fox (Ed.), *Women and mathematics: Research perspectives for change*. NIE Papers in Education and Work, No. 8. Washington, DC: National Institute of Education.

Fredericksen, N. (1984). The real test bias: Influences of testing in teaching and learning. *American Psychologist*, **39**, 193-202.

Freeman, C. Hawkins, J. & Char, C. (1984). *Information management tools for classrooms: Exploring data management systems*. Technical Report No. 28. New York: Bank Street College of Education, Center for Children and Technology.

Friendly, M. (1988). *Advanced Logo: A language for learning*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Gagne, R.M. (1970). *The conditions of learning*. New York: Holt, Rinehart & Winston.

Gentner, D. & Stevens, A.L. (Eds). (1983). *Mental models*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Glaser, R. (1984). Education and thinking: The role of knowledge. *Americal Psychologist*, **39**, 93-104.

Goldenberg, E.P. (1988). Mathematics, metaphors, and human factors: Mathematical, technical and pedagogical challenges in the educational use of graphical representation and functions. *Journal of Mathematical Behavior*, 7, 135-173.

Goldstein, I. & Papert, S. (1977). Artificial intelligence, language and the study of knowledge. *Cognitive Science*, **1**, 84-123.

Goody, J. (1977) *The domestication of the savage mind*. New York: Cambridge University Press.

Goody, J. (1987). *The interface between the written and the oral*. Cambridge: Cambridge University Press.

Gorman, H. & Bourne, L.E. (1983). Learning to think by learning LOGO: Rule learning in third grade compnter programmers. *Bulletin of the Psychonomic Society*, **21**, 165-167.

Gould, S.J. (1980). Senseless signs of history. In S.J. Gould (Ed.), *The panda's thumb: More reflections in natural history*. New York: W.W. Norton.

Grace, N. & Cassidy, J. (1990). *Years 1-10 mathematics sourcebook: Years 9/10 (Draft)*. Brisbane: Queensland Education Department, Curriculum Development Services.

Greenfield, P.M. (1966). On culture and conversation. In J.S. Bruner (Ed.), *Studies in cognitive growth*. New York: John Wiley, 1966.

Greenfield, P.M. (1984). *Mind and media*. Cambridge, MA: Harvard University Press.

Greeno, J.G. (1985). Looking across the river: Views from the two banks of research and development iu problem solving. In J.W. Segal, S.F. Chipman & R. Glaser (Eds), *Thinking and learning skills*. Vol. 1: Relating instruction to research. Hillsdale, NJ: Lawrence Erlbaum Associates

Greeno, J.G. & Simon, H.A. (1988). Problem solviug and reasoning. In R.C. Atkinson, R. Herrnstein, G. Lindzey & R.D. Luce (Eds), *Stevens' handbook of experimental psychology* (Rev. ed.). New York: Wiley.

Haertel, G.D. (1978). *Literature of early adolescence and implications for programming*. Washington, DC: US Government Printing Office.

Haigh, R.W. (1985). Planning for computer literacy. *Journal of Higher Education*, **56**, 161-171.

Hamolsky, M.C. (1983). Effective teacher training programs: Of what do they consist? A review of the research. Paper presented at the annual meeting of the Association for Educational Data Systems, Portland, Oregon, May 9-13.

Handley, H.M. & Morse, L.W. (1984). Two-year study relating adolescent self concept and gender role perceptions to achievement and attitudes toward science. *Journal of Research in Science Teaching*, **21**, 599-607.

Harper, D. (1989). *Logo: Theory and practise*. Pacific Grove, CA: Brooks/Cole Publishing Co.

Harre, R. (1984). *Personal being*. Cambridge, MA: Harvard University Press.

Harre, R. & Madden, E.H. (1975). *Causal powers: A theory of natural necessity*. Oxford: Basil Blackwell.

Hattie, J. (1988). *Golden opportunities for learning, control and literacies*. Unpublished report financially supported by the Bernard van Leer Foundation and the Australian Computer Society.

Hattie, J. & Fitzgerald, D. (1988). Sex differences in attitudes, achievement and use of computers. *Australian Journal of Education*, **31**, 3-26.

Hawkins, J. (1983). *Learning Logo together: The social context*. Technical report No. 13. New York: Bank Street College of Education, Center for Children and Technology.

Hawkins, J. (1987). Computers and girls: Rethinking the issues. In R.D. Pea & K. Sheingold (Eds), *Mirrors of minds: Patterns of experience in educational computing*. Papers from the Center for Children and Technology, Bank Street College. Norwood, NJ: Ablex Publishing Co.

Hawkins, J. Char, C. & Freeman, C. (1984). *Software tolls in the classroom*. Report to the Carnegie Corporation. New York: Bank Street College of Education, Center for Children and Technology.

Hawkins, J. Mawby, R. & Ghitman, M. (1987). Practises of novices and experts in critical inquiry. In R.D. Pea & K. Sheingold (Eds), *Mirrors of minds: Patterns of experience in educational computing*. Papers from the Center for Children and Technology, Bank Street College of Education. Norwood, NJ: Ablex.

Hawkins J. & Sheingold, K. (1987). The beginning of a story: Computers and the organization of learning in classrooms. In J. Culbertsou & L.L. Cunningham (Eds), *Uses of microcomputers and other technology in education*. Chicago, IL: University of Chicago Press.

Hawkins, J., Sheingold, K., Gearhart, M. & Berger, C. (1982). Microcomputers in schools: Impact on the social life of elementary classrooms. *Journal of Applied Developmental Psychology*, **3**, 361-373.

Hayes-Roth, B. & Hayes-Roth, F. (1979). A cognitive model of planning. *Cognitive Science*, **3**, 275-310.

Hayes-Roth, F., Waterman, D. & Lenat, D. (Eds). (1984). *Building expert systems*. Reading, MA: Addison-Wesley.

Hayman, J., Rayder, N., Stenner, A.J. & Madey, D. (1979). On aggregation, generalization and utility in educational evaluation. *Educational Evaluation and Policy Analysis*, **1**, 31-39.

Heid, M.K. (1988). Resequencing skills and concepts in applied calculus using the computer as a tool. *Journal of Research in Mathematics Education*, **19**, 3-25.

Heid, M.K. (1989). How symbolic mathematical systems could and should affect precollege mathematics. *Mathematics Teacher*, **82**, 410-419.

Heller, J.I. & Greeno, J.G. (1979). Information processing analyses of mathematical problem solving. In R. Tyler & S. White (Eds). *Testing, teaching and learning*. Washington, DC: National Institute of Education.

Heppell, S. (1989). Today's students, tomorrow's schools, next decade's technology: Defining the issues and seeking solutions for teacher education. Paper read at the CAL '89 Conference held at the University of Surrey, Guildford.

Herrnstein, R.J., Nickerson, R.S., de Sanchez, M. & Swets, J.A. (1986). Teaching thinking skills. *American Psychologist*, **41**, 1279-1289.

Hess, R. & Miura, I. (1983). Gender and socioeconomic differences in enrolled computer camps and classes. Unpublished manuscript. Stanford, CA: Stanford University.

Hollan, J.D., Hutchins, E. & Weitzman, L. (1984). STEAMER: An interactive inspectable simulation-based training system. *AI Magazine*, **5**, 15-27.

Hoyles, C. (1988). *Girls and computers: General issues and case studies of Logo in the mathematics classroom*. Bedford Way Paper 34. London: University of London, Institute of Education.

Hoyles, C., Sutherland, R. & Evans, J. (1986). Using Logo in the mathematics classroom. What are the implications for pupil devised goals? *Computers in Education*, **12**, 61-73.

Hughes, M. Brackenridge, A. & Macleod, H. (1987). Gender and social interaction in early Logo use. In J.H. Collins, N. Estes, W.D.Gattis & D. Walker (Eds), *The Sixth International Conference on Technology and Education*. Vol. 1. Edinburgh: CEP.

Hughes, M. & Greenhough, P. (1989). Gender and social interaction in early Logo use. In J.H. Collins, N. Estes, W.D. Gattis & D. Walker (Eds), *The Sixth International Conference on Technology and Education*. Vol. 1. Edinburgh: CEP.

Hughes, M. & Macleod H. (1986). Using Logo with very young children. In R.W. Lawler, B. Du Boulay, M. Hughes & H. Macleod (Eds), *Cognition and computers: Studies in learning*. Chichester: Ellis Horwood.

Hunter, B. (1983). *My students use computers: Learning activities for computer literacy*. Reston: Reston Publishing Co.

Husen, T. & Postlethwaite, T.N. (Eds). (1985). *The International Encyclopaedia of Education: Research and studies*. Vol. 2C. Oxford: Pergamon Press.

Jackson, A., Fletcher, B. & Messer, D.J. (1986). A survey of microcomputer use and provision in primary schools. *Journal of Computer Assisted Learning*, **2**, 45-55.

Jeffries, R., Turner, A.A., Polson, P.G. & Atwood, M.E. (1981). The processes involved in designing software. In J. Anderson (Ed.), *Cognitive skills and their acquisition*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Johnson, D.C., Anderson, R.E. Hansen, T.P. & Klassen, D.L. (1980). Computer literacy - what is it? *The Mathematics Teacher*, **73**, 91-96.

Johnson, D.W. & Johnson, R.T. (1989). *Cooperation and competition: Theory and research*. Edina, MN: Interaction Book Company.

Kaufman, A.S. & Kaufman, N.L. (1983). *Kaufman Assessment Battery for Children (K-ABC)*. Circle Pines, MN: American Guidance Service.

Keisler, S., Sproull, L. & Eceles, J. (1988). Pool halls, chips and war games in the culture of computing. *Psychology of Women Quarterly*, **9**, 451-462.

Kirchner, G., Martin, D. & Johnson, C. (1983). Simon Fraser University videodisc project: Part II: Field testing an experimental videodisc with elementary school children. *Videodisc/Videotex*, **3**, 45-58.

Klahr, D. & Carver, S.M. (1988). Cognitive objectives in a Logo debugging curriculum: Instruction, learning and transfer. *Cognitive Psychology*, **20**, 362-404.

Kozulin, A. (1986). The concept of activity in Soviet psychology. *American Psychologist*, **41**, 264-274.

Kreinberg, N. (1981). 1000 teachers later: Women, mathematics and the components of change. *Public Affairs Report*, **22**, 1-7.

Kull, J.A. (1986). A Brunerian approach to teaching and learning Logo. Paper presented at the Annual Meeting of the American Educational Research Association. San Francisco, CA.

Kunkle, D. & Burch, C.I., Jr. (1984). Symbolic computer algebra. *Mathematics Teacher*, **77**, 209-214.

Kurland, D.M. (Ed.) (1984). *Symposium: Developmental studies of computer programming skills*. Tech. Report No. 29. New York: Bank Street College of Education, Center for Children and Technology.

Kurland, D.M. (1987). *The Bank Street Writer*. New York: Bank Street College of Education, Center for Children and Technology.

Kurland, D.M., Clement, C.A., Mawby, R. & Pea, R.D. (1987). Mapping the cognitive demands of learning to program. In D.N. Perkins, J. Lochhead & J. Bishop (Eds), *Thinking: The second international conference*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Kurland, D.M., Mawby, R. & Cahir, N. (1984). *The development of programming expertise*. Paper presented at the meeting of the American Educational Research Association, New Orleans, LA.

Kurland, D.M. & Pea, R.D. (1985). Children's mental models of recursive Logo programs. *Journal of Educational Computing Research*, **1**, 235-243.

Larkin, J.H., McDermott, J., Simon, P.P. & Simon, H.A. (1980). Expert and novice performance in solving physics problems. *Science*, **208**, 1335-1342.

Lawler, R.W. (1985). *Computer experience and cognitive development*. Chichester: Ellis Horwood.

Lawton, J. & Gerschner, V.T. (1982). A review of the literature on attitudes towards computers and computerized instruction. *Journal of Research and Development in Education*, **16**, 50-55.

Lenney, E. (1977). Women's self-confidence in achievement settings. *Psychological Bulletin*, **84**, 1-13.

Leont'ev, A.N. (1981). *Problems in the development of the mind*. Moscow: Progress Publishers.

Lepper, M.R. (1982). *Microcomputers in education: Motivational and social issues*. Palo Alto, CA: Stanford University, Department of Psychology.

Lepper, M.R. (1985). Microcomputers in education: Motivational and social issues. *American Psychologist*, **40**, 1-18.

Leron, U. (1985). LOGO today: Vision and reality. *The Computing Teacher*, **12**, 26-32.

Levin, J., Boruta, M. & Vasconcellos, M. (1983). Microcomputer-based environments for writing: A writer's assistant. In A.C. Wilkinson (Ed.), *Classroom computers and cognitive science*. New York: Academic Press.

Levin, J. & Fowler, H.S. (1984). Sex, grade and course differences in attitudes that are related to cognitive performance in secondary science. *Journal of Research in Science Teaching*, **21**, 151-166.

Light, P.H. & Colbourn, C.J. (1987). The role of social processes in children's microcomputer use. In W.A. Kent & R. Lewis (Eds), *Computer-assisted learning in the social sciences and humanities*. Oxford: Basil Blackwell.

Linn, M.C. (1985). Fostering equitable consequences from computer learning environments. *Sex Roles*, **13**, 229-240.

Linn, M.C. & Dalbey, J. (1985). Cognitive consequences of programming instruction: Instruction, access and ability. *Educational Psychologist*, **20**, 191-206.

Lipkin, J. (1982). The troubling equity issue in computer education: Is computer literacy the answer or the problem? *Education Times*, October 25.

Lochhead, J. (1985). Teaching analytic reasoning skills through pair problem solving. In J.W. Segal, S.F. Chipman & R. Glaser (Eds), *Thinking and learning skills*. Vol. 1: Relating instruction to research. Hillsdale, NJ: Lawrence Erlbaum Associates.

Lockheed, M., Nielsen, A. & Stone, M. (1983). Sex differences in microcomputer literacy. Paper presented at the National Educational Computer Conference. Baltimore, MD.

Longworth, N. (1981). We're moving into the information society. What shall we teach children? *Computer Education*, **38**, 17-19.

Loop, L. & Christensen, P. (1982). Exploring the microcomputer learning environment. Report No. 5, Far West Laboratory for Educational Research and Development.

Luehrman, A. (1980). Should the computer teach the student, or vice-versa? In R.P. Taylor (Ed.), *The computer in the school: Tutor, tool, tutee*. New York: Teachers College Press.

Luehrman, A. (1981). Computer literacy - what should it be? *Mathematics Teacher*, **74**, 682-686.

Luria, A.R. (1976). *Cognitive development: Its cultural and social foundations*. Cambridge, MA: Harvard University Press.

Luria, A.R. (1979). *The making of mind: A personal account of Soviet psychology*. Cambridge, MA: Harvard University Press.

Maccoby, E.E. & Jacklin, C.M. (1974). *The psychology of sex differences*. Palo Alto, CA: Stanford University Press.

Maier, N.R.F. & Casselman, G.G. (1970). The SAT as a measure of problem solving ability in males and females. *Psychological Reports*, **26**, 927-939.

Martin, L.M.M. & Scribner, S. (1991). Laboratory for Cognitive Studies of Work: A case study of the intellectual implications of a new technology. *Teachers College Record*, **92**, 582-602.

Maurer, S.B. (1984a). College entrance mathematics in the year 2000. *Mathematics Teacher*, **77**, 422-428.

Maurer, S.B. (1984b). The effects of a new college mathematics curriculum on high school mathematics. In A. Ralston & G.S. Young (Eds), *The future of college mathematics*. New York: Springer Verlag.

Mawby, R., Clement, C.A., Pea, R.D. & Hawkins, J. (1984). *Structured interviews on children's conceptions of computers*. Technical Report No. 19. New York: Bank Street College of Education, Center for Children and Technology.

McCarthy, D. (1978). *McCarthy Screening Test*. New York: Psychological Corporation.

Miller, G.E. & Emihovich, C. (1986). The effects of mediated programmed instruction on preschool childrens' self-monitoring. *Journal of Educational Computing Research*, **2**, 283-297.

Miller, S.M. (1980). Why having control reduces stress: If I can stop the roller coaster, I don't want to get off. In J. Garber & M.E.P. Seligman (Eds), *Human helplessness: Theory and applications*. New York: Academic Press.

Milner, S. (1973). The effects of computer programming on performance in mathematics. Paper presented at the annual meeting of the AERA, New Orleans, LA, February, 1973.

Minnesota Educational Computing Consortium (1979). *Computer literacy study*. St Paul, MN.: Minnesota Educational Computing Consortium.

Minsky, M. (1970). Form and content in computer science. *Communications of the ACM*, **17**, 197-215.

Minsky, M. (1983). Why people think computers can't. *Technology Review*, **86**, 65-81.

Minuchin, P. & Shapiro, E.K. (1983). The school as a context for social development. In E.M. Hetherington (Ed.), *Socialization, personality, and social development* (Vol. IV, of P.H. Mussen (Ed.), *Handbook of child psychology* (4th ed.)). New York: Wiley.

Miura, I & Hess, R. (1983). Sex differences in computer access, interest and usage. Paper presented at the meeting of the APA, Anaheim, CA.

Moont, S. (1984). Will women be the drop-outs of the information age? Paper presented at the Second Australian Computers and Education Conference, August 1984.

Nanson, C. (1982). Teaching computer use - not programming. *Electronic Learning*, **2**, 24-31.

National Commission on Excellence in Education (1983). *A nation at risk: The imperative for educational reform*. Washington, DC: National Commission on Excellence in Education.

National Council of Teachers of Mathematics (1980). *An agenda for action: Recommendations for school mathematics of the 1980s*. Palo Alto, CA: Dale Seymour.

National Education Association (1983). *A teacher survey: Computers in the classroom*. Washington, DC: National Education Association.

National Science Board Commission on Precollege Education in Mathematics, Science and Technology (1983). *Educating Americans for the 21st century*. Washington, DC: National Science Foundation.

Neisser, U. (1976). *Cognition and reality: Principles and implications of cognitive psychology*. San Francisco: W.H. Freeman.

New South Wales Department of Education Computer Education Unit (1985). *An evaluator's guide to educational software and related materials*. Sydney: NSW Department of Education.

Newell, A. & Simon, H.A. (1972). *Human problem solving*. Englewood Cliffs, NJ: Prentice Hall.

Newman, D., Riel, M.R. & Martin, L. (1983). Cultural practises and Piaget's theory: The impact of a cross-cultural research program. In D. Kuhn & J.A. Meacham (Eds), *On the development of developmental psychology*. Basel: Karger.

Nickerson, R.S. (1982). Computer programming as a vehicle for teaching thinking skills. *Thinking, The Journal of Philosophy for Children*, **4**, 42-48.

Nickerson, R.S., Perkins, D.N. & Smith, E.E. (1985). *The teaching of thinking*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Niemiec, R.P. & Walberg, H.J. (1991). The effects of computers on learning. *International Journal of Educational Research*, **17**, 99-121.

Noble, D. (1985). Computer literacy and ideology. In D. Sloan (Ed.), *The computer in education: A critical perspective*. New York: Teachers College Press.

Noyelle, T. (1984). *Working in a world of high technology: Employment problems and mobility prospects of disadvantaged workers*. New York: Columbia University.

Office of Technology Assessment (1982). *Informational technology and its impact on American education*. Washington, DC: US Government Printing Office.

Ohlsson, S. (1986). Some principles of intelligent tutoring. *Instructional Science*, **14**, 293-326.

Olson, D.R. (1976). Culture, technology and intellect. In L.B. Resnick (Ed.), *The nature of intelligence*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Olson, D.R. (1977). From utterance to text: The bias of language in speech and writing. *Harvard Educational Review*, **47**, 257-281.

Olson, D.R. (1985). Computers as tools of the intellect. *Educational Researcher*, **14**, 5-7.

Olson, D.R. (1988). *Schoolworlds/Microworlds*. New York:Pergamon Press.

Olson, D.R. & Bruner, J.S. (1974). Learning through experience and learning through media. In D.R. Olson (Ed.), *Media and symbols: The forms of expression, communication, and education*. Chicago: University of Chicago Press.

Ong, W.J. (1982). *Orality and literacy: The technologizing of the word*. New York: Methuen.

Osen, L. (1974). *Women in mathematics*. Cambridge, MA: MIT Press.

Page, B. & Wallig, C. (1983). Staff training for successful CAI. Paper presented at the annual meeting of AEDS, Portland, Oregon.

Paisley, W.J. (1983). Computerizing information: Lessons from the videotex trial. *Journal of Communication*, **33**, 152-161.

Paisley, W.J. & Chen, M. (1982). *Children and electronic text: Challenges and opportunities for the 'new literacy'*. Palo Alto, CA: Stanford University: Institute for Communications.

Papert, S. (1972a). Teaching children thinking. *Programmed Learning and Educational Technology*, **9**, 245-255.

Papert, S. (1972b). Teaching children to be mathematicians versus teaching about mathematics. *International Journal for Mathematical Education, Science and Technology*, **3**, 249-262.

Papert, S. (1980). *Mindstorms: Children, computers and powerful ideas*. Brighton: Harvester Press.

Papert, S. (1987a). Computer criticism vs. technocentric thinking. *Educational Researcher*, **16**, 22-30.

Papert, S. (1987b). A critique of technocentrism in thinking about the school of the future. In B. Sendov & I. Stanchev (Eds), *Children in the information age: Opportunities for creativity, innovation and new activities*. Selected papers from the second international conference, Sofia, Bulgaria, 19-23 May 1987.

Papert, S., Watt, D. diSessa, A. & Weir, S. (1979). *Final report of the Brookline LOGO project*. LOGO Memo No. 53. Cambridge, MA: MIT Artificial Intelligence Laboratory.

Parsons, J., Kaczala, C. & Muce, J. (1982). Socialization of achievement attitudes and beliefs: Classroom influences. *Child Development*, **53**, 322-339.

Pea, R.D. (1982). What is planning development the development of? In D. Forbes & M. Greenberg (Eds), *New directions in child development: Children's planning strategies*. San Francisco: Jossey-Bass.

Pea, R.D. (1985). Integrating human and computer intelligence. In E.L. Klein (Ed.), *New directions for child development. No. 8. Children and computers*. San Francisco: Jossey-Bass.

Pea, R.D. (1987). The aims of software criticism: Reply to Professor Papert. *Educational Researcher*, **16**, 4-8.

Pea, R.D., Hawkins, J., Clement, C.A. & Mawby, R. (1984). *The development of expertise in Logo by children*. New York: Bank Street College of Education, Center for Children and Technology.

Pea, R.D., Hawkins, J. & Sheingold, K. (1983). Developmental studies on learning LOGO computer programming. Paper presented at the biennial meeting of the Society for Research in Child Development, Detroit, MI, April 1983.

Pea, R.D. & Kurland, D.M. (1983). *Logo programming and the development of planning skills*. Technical Report No. 16. New York: Bank Street College of Education, Center for Children and Technology.

Pea, R.D. & Kurland, D.M. (1984). On the cognitive effects of learning computer programming. *New Ideas in Psychology*, **2**, 137-167.

Pea, R.D. & Kurland, D.M. (1987a). Cognitive technology for writing. *Review of Research in Education*, **14**, 277-326.

Pea, R.D. & Kurland, D.M. (1987b). Logo programming and the development of planning skills. In K.Sheingold & R.D. Pea (Eds), *Mirrors of minds: Patterns in experience in educational computing*. Norwood, NJ: Ablex.

Pea, R.D., Kurland, D.M. & Hawkins, J. (1985). Logo and the development of thinking skills. In M. Chen & W. Paisley (Eds), *Children and microcomputers: Formative studies*. Beverly Hills, CA: Sage.

Pea, R.D., Kurland, D.M. & Hawkins, J. (1987). Logo and the teaching of thinking skills. In R.D. Pea & K. Sheingold (Eds), *Mirrors of minds: Patterns in experience in educational computing*. Papers from the Center for Children and Technology, Bank Street College of Education. Norwood, NJ: Ablex.

Pea, R.D. & Sheingold, K. (Eds). (1987). *Mirrors of minds: Patterns of experience in educational computing*. Papers from the Center for Children and Technology, Bank Street College of Education. Norwood, NJ: Ablex.

Piaget, J. (1952). *The origins of intelligence in children*. New York: International Universities Press.

Piaget, J. (1954). *The construction of reality in the child*. New York: Basic Books.

Piaget, J. (1970). Piaget's theory. In P.H. Mussen (Ed.), *Carmichael's manual of child psychology*. Vol. 1. 3rd edition. New York: Wiley.

Piaget, J. (1972). *The psychology of intelligence*. Totowa, NJ: Littlefield, Adams.

Piaget, J. (1973). To understand is to invent. New York: Grossman.

Poirot, J.L. (1980). *Computers and education*. Austin, Texas: Stirling Swift.

Polya, G. (1957). *How to solve it: A new aspect of mathematical method* (2nd ed.). Garden City, NY: Doubleday-Anchor.

Queensland Education Department (1988). *Practical computer methods: Guidelines*. Brisbane: Queensland Education Department.

Queensland Institute of Technology, Educational Research and Development Unit (1985). *ERDU/NEWTECH/HITECH/INFTECH GLOSSARY*. Brisbane: Queensland Institute of Technology.

Raaheim, K. (1974). *Problem solving and intelligence*. Oslo/Bergen: Universitetsforlaget.

Resnick, L.B. (1983). Mathematics and science learning: A new conception. *Science*, **220**, 477-478.

Rice, R.E. (1984). *The new media: Communication, research and technology*. Beverly Hills, CA: Sage.

Ridgway, J. (1988). Of course ICAI is impossible . . ., Worse though, it might be seditious. In J. Self (Ed.), *Artificial intelligence and human learning: Computer-aided instruction*. London: Chapman & Hall.

Ridgway, J. & Passey, D. (1991). A constructivist approach to educational computing. *Australian Educational Computing*, **6**, 4-9.

Robinson, M.A. & Uhlig, G.E. (1988). The effects of guided discovery Logo instruction on mathematical readiness, visual motor integration and attendance of first grade students. *Journal of Human Behavior and Learning*, **5**, 1-13.

Roblyer, M.D. (1981). Instructional design versus authoring of courseware: Some critical differences. *AEDS Journal*, **14**, 173-181.

Rogoff, B. & Lave, J. (1986). *Everyday cognition*. Cambridge, MA: Harvard University Press.

Ross, P. & Howe, J. (1981). Teaching mathematics through programming: Ten years on. In R. Lewis & D. Tagg (Eds), *Computers in education*. Amsterdam: North Holland.

Rowe, H.A.H. (1985). *Problem solving and intelligence*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Rowe, H.A.H. (1988). Metacognitive skills: promises and problems. *Australian Journal of Reading*, **11**, 227-238.

Rowe, H.A.H. (1989). Teach learning strategies. *SET*, **1**, Number 14 (whole number).

Rowe, H.A.H. (1991a). *Intelligence: Reconceptualization and measurement*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Rowe, H.A.H. (1991b). Learning with computers: What type of research? *Proceedings* of the Ninth Australian Computers in Education Conference. ACEC, Brisbane, September 1991.

Rubin, A.D. & Bruce, B.C. (1988). QUILL: Reading and writing with a microcomputer. In B.A. Hutson (Ed.), *Advances in reading and language research*. Greenwich, CT: JAI Press.

Ryan, M. (1991). *The Queensland Sunrise Centre: A report of the first year*. Hawthorn, Vic: Australian Council for Educational Research.

Salomon, G. (1988). Artificial intelligence and natural wisdom: How cultural artifacts can cultivate the mind. Keynote address presented at the 24th International Congress of Psychology, Sydney, Australia.

Schieffelin, B.B. & Gilmore, P. (1986). *The acquisition of literacy: Ethnographic perspectives*. Norwood, NJ: Ablex.

Schoenfeld, A.H. (1985). *Mathematical problem solving*. Orlando: Academic Press.

Scott, K.P. (1984). Effects of an intervention on middle school pupils' decision making, achievement and sex role flexibility. *Journal of Educational Research*, **77**, 369-375.

Scribner, S. & Cole, M. (1981). *The psychology of literacy*. Cambridge, MA: Harvard University Press.

Segal, J.W., Chipman, S.F. & Glaser, R. (Eds). (1985). *Thinking and learning skills*. Vol. 1: Relating instruction to research. Hillsdale, NJ: Lawrence Erlbaum Associates.

Sharples, M. (1985). *Cognition, computers and writing*. Chichester: Ellis Horwood.

Shavelson, R.J. (1979). Applications of cluster analysis in educational research: Looking for a needle in a haystack. *British Educational Research Journal*, **5**, 45-53.

Shavelson, R.J. & Stern, P. (1981). Research on teachers' pedagogical thoughts, judgments, decisions and behavior. *Review of Educational Research*, **51**, 455-498.

Sheil, B.A. & Masinter, L.M. (Eds). (1983). *Papers on Interlisp-D*. Palo Alto, CA: Xerox Palo Alto Research Center.

Sheingold, K., Hawkins, J. & Char, C. (1984). I'm the thinkist, you're the typist: The interaction of technology and the social life of classrooms. *Journal of Social Issues*, **40**, 49-61.

Sheingold, K., Kane, J. & Endreweit, M. (1983). Microcomputer use in schools: Developing a research agenda. *Harvard Educational Review*, **53**, 412-432.

Sheingold, K., Kane, J., Endreweit, M. & Billings, K. (1981). *Study of issues related to implementation of computer technology in schools* (Tech. Rep. No. 2). New York: Bank Street College of Education, Center for Children and Technology.

Sherman, M.T. (1983). *Computers in education: A report*. Concord, MA: Bates Publishing Company.

Shulman, L.S. & Elstein, A.S. (1975). Studies of problem solving, judgment, and decision making. In F.N. Kerlinger (Ed.), *Review of research in education*. Vol. 3. Itaska, IL: F.E. Peacock.

Siann, G., Durndell, A., Macleod, H. & Glissow, P. (1988). Stereotyping in relation to the gender gap in participating in computing. *Educational Research*, **30**, 98-103.

Siann, G. & Macleod, H. (1986). Computers and children of primary school age: Issues and questions. *British Journal of Educational Technology*, **17**, 133-144.

Simon, H.A. (1977). *Models of discovery*. Dordrecht, Holland: Reidel.

Simon, H.A. (1980). Problem solving in education. In D.T. Tuma & R. Reif (Eds), *Problem solving and education: Issues in teaching and research*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Simon, H.A. (1987). The steam engine and the computer: What makes technology revolutionary? *EDUCOM Bulletin*, **22**, 2-5.

Slavin, R.E. (1985). An introduction to cooperative learning research. In R. Slavin (Ed.), *Learning to cooperate, learning to learn*. New York: Plenum Press.

Slavin, R.E. (1986). Small group methods. In M. Dunkin (Ed.), *The International Encyclopaedia of Teaching and Teacher Education*. New York: Pergamon Press.

Sleeman, D. & Brown, J.S. (Eds). (1982). *Intelligent tutoring systems*. New York: Academic Press.

Smith, L. (1989). Changing perspectives in developmental psychology. In C. Deforges (Ed.), *Early childhood Education*. Educational Monograph Series, No. 4. Edinburgh: Scottish Academic Press.

Snow, R.E. & Yallow, E. (1982). Education and intelligence. In R. Sternberg (Ed.), *Handbook of human intelligence*. Cambridge: Cambridge University Press.

Soloway, E. (1988). It's 2020: Do you know what your children are learning in programming class? In R.S. Nickerson & P.P. Zodhiates (Eds), *Technology in education: Looking toward 2020*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Soloway, E. & Ehrlich, K. (1984). Empirical studies of programming knowledge. *IEEE Transactions on Software Engineering*, **SE-10**, 595-609.

Soloway, E., Ehrlich, K., Bonar, J. & Greenspan, J. (1982). What do novices know about programming? In B. Shneiderman & A. Badre (Eds), *Directions in human-computer interactions*. Norwood, NJ: Ablex.

Soloway, E. Lochhead, J. & Clement, J. (1982). Does computer programming enhance problem solving ability? Some evidence on algebra word problems. In R. Seidel, R. Anderson & B. Hunter (Eds), *Computer literacy*. New York: Academic Press.

Sommerville, A. (1982). *Software design*. London: Addison-Wesley.

Steen, A.L. & Albers, D.J. (1981). *Teaching teachers, teaching students*. Boston: Birkhauser.

Steinkamp, M.W. & Maehr, M.L. (1984). Gender differences in motivational orientations toward achievement in school science. *American Educational Research Journal*, **21**, 39-59.

Taylor, R.P. (Ed.). (1980). *The computer in the school: Tutor, tool, tutee*. New York: Teachers College Press.

Torrance, E.P. (1974). *Torrance Tests of Creative Thinking*. Washington: Personnel Press.

Tufte, E.R. (1983). *The visual display of quantitative information*. Cheshire, CT: Graphics Press.

Turkle, S. (1984). *The intimate machine*. New York: Simon & Schuster.

Underwood, J. (1986). The role of the computer in developing children's classificatory abilities. *Computers in Education*, **10**, 175-180.

Underwood, J. (1989). The effectiveness of computer-based learning in developing children's classificatory abilities. In R.C. King & J.K. Collins (Eds), *Social applications and issues in psychology*. North Holland: Elsevier.

Underwood, J. & Underwood, G. (1990). *Computers and learning: Helping children acquire thinking skills*. Oxford: Basil Blackwell.

VanLehn, K. (1983). *Felicity conditions for human skill acquisition: Validating an AI-based theory* (Tech. Report No. CIS-21). Palo Alto, CA: Xerox Palo Alto Research Center.

VanLehn, K. (1985). *Theory reformulation caused by an argumentation tool*. Palo Alto, CA: Xerox Palo Alto Research Center.

Vygotsky, L.S. (1962). *Thought and language*. Cambridge, MA: MIT Press.

Vygotsky, L.S. (1978). *Mind in society: The development of higher psychological processes*. Cambridge, MA: Harvard University Press.

Walberg, H.J. (1976). Psychology of learning environments: Behavioral, structural or perceptual? In L.S. Shulman (Ed.), *Review of research in education*. Vol. 4. Itaska, IL: F.E. Peacock.

Walker, D.F. (1987). Logo needs research: A response to Papert's paper. *Educational Researcher*, **16**, 9-11.

Watt, D.H. (1982). Logo in the schools. *Byte*, 7, 116-134.

Webb, N.M. (1980). Group process and learning in an interacting group. *The Quarterly Newsletter of the Laboratory of Comparative Human Cognition*, **2**, 10-15.

Webb, N.M. (1984). Microcomputer learning in small groups: Cognitive requirements and group processes. *Journal of Educational Psychology*, **76**, 1076-88.

Weinstein, C.E. & Underwood, V.L. (1985). Learning strategies: The *how* of learning. In J.W. Segal, S.F. Chipman & R. Glaser (Eds), *Thinking and learning skills*. Vol. 1: Relating instruction to research. Hillsdale, NJ: Lawrence Erlbaum Associates.

Weizenbaum, J. (1976). *Computer power and human reason*. San Francisco: W.H. Freeman & Co.

Wertsch, J.V. (1985). *Vygotsky and the social formation of mind*. Cambridge, MA: Harvard University Press.

Wessel, M. (1974). *Freedom's edge: The computer threat to society*. Reading, MA: Addison-Wesley.

Whimbey, A. & Lochhead, J. (1980). *Problem solving and comprehension: A short course in analytical reasoning* (2nd ed.). Philadelphia: Franklin Institute Press.

Whipple, W.R. (1987). Collaborative learning: Recognizing it when we see it. *AAHE Bulletin*, October 1987, 4-6. ERIC Microfiche: ED 289398, 1988.

White, S. (1981). The new liberal arts. In J.D. Koerner (Ed.), *The new liberal arts*. New York: Alfred P. Sloan Foundation.

Widmer, C. & Parker, J. (1983). Needed: Less computer apprehension; Wanted: Computer enthusiasts; Found: In-service techniques. 21st Annual Convention Proceedings, Association for Educational Data Systems. Washington, DC.

Wilf, H.S. (1982). The disk with the college education. *American Mathematical Monthly*, **89**, 4-8.

Williams, F. & Williams, V. (1984). *Microcomputers in elementary education: Perspectives and implementation*. Belmont, CA: Wadsworth.

Winkle, L.W. & Mathews, W.M. (1982). Computer equity comes of age. *Phi Delta Kappa*, **63**, 314-315.

Wirth, N. (1973). *Systematic programming: An introduction*. Englewood Cliffs, NJ: Prentice-Hall.

Wood, D.J., Bruner, J.S. & Ross, G. (1976). The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry*, **17**, 89-100.

Author Index

Subject Index