2003

Assessment and evaluation of computer science education

Jennifer R. Parham

The University of Montana

AN ASSESSMENT AND EVALUATION OF COMPUTER SCIENCE EDUCATION

by

Jennifer R. Parham

B.S. Appalachian State University, NC 1999

presented in partial fulfillment of the requirements

for the degree of

Master of Science

The University of Montana

December 2003

Approved by:

[illegible]

Chairperson

[illegible]

Dean, Graduate School

Date 08 August 2003

Parham, Jennifer R. M.S., 2003 Computer Science

An Assessment and Evaluation of Computer Science Education

Chairperson of the Supervisory Committee: Dr. Don Morton [illegible]

Thesis Statement: This research shows an audience of fellow computer science teachers that there is benefit in testing college student's scientific reasoning skills and that the students increase their logical thinking skills in computer science classes when they are presented with real world problems, engaged in group activities, and given hands-on experiments.

Students' scientific reasoning skills from several different computer science courses are analyzed and tested before and after taking the computer science course to measure their improvement in logical thinking. The higher order thinking skills test results are compared, and the overall class and individual student improvement between the classes that use different teaching strategies are analyzed. Also, the test results are compared with students' ability to perform adequately in a computer science course. This information is valuable to advisors when placing students in computer science courses.

In the assessments of students' entering knowledge and evaluations of students' thinking improvement, the students' individual higher order thinking skills as well as their overall level of cognitive development are analyzed. Through the analysis of their individual skills and level of thinking, one can expand conclusions about the students' and teacher's performance over time.

PREFACE

I want to start this research paper by explicitly stating what I intend the audience to gain from reading this paper and which ideas presented in this paper I feel are the most important. Firstly, I want the audience to understand there is a problem with the way we are educating our future computer scientists, and there needs to be some movement toward an active learning approach to the discipline of computer science, just as there is with other sciences. Also, if we plan to change our current style of computer science education, then there must be some type of assessment of our students' learning. My definition of learning is not regurgitating information from a book or a professor's mouth about a specific topic in a discipline. But rather, learning is gaining a richer understanding of a discipline, sparking an interest in an area where there wasn't any previous interest, and developing skills for a student to progress further by becoming a self-directed learner in a particular discipline.

Innovative teaching and the assessment of learning are what motivated me to conduct this research for my thesis, and these are the two key points I want the audience to take home with them. Primarily, my motivations for this thesis stem from my own personal experience of being a computer science student and the love I have for teaching. I have a philosophy that any student can learn any subject given that the subject is presented in such a way the student is able to benefit from the style of teaching. In my computer science education, I have seen very few different methods of presenting the material in a classroom environment, other than the rote lecturer. However, I have learned material through lecturing, hands on experience, and peer interaction throughout my experience in computer science.

Our current computer science education has left myself and many other students unprepared when we leave the academic realm and step into the real world. This is what has led me to exploration of other ways of presenting computer science material in a classroom environment. Therefore, I assessed the effects a new teaching method in a computer science course had on the students' logical thinking skills. This assessment shows promising results that demonstrate a need to continue research in computer science education.

Jennifer R. Parham

ACKNOWLEDGMENTS

I want to thank Dr. Mark Cracolice for providing me with logical thinking assessment tests. Also, I want to say thanks to my computer science class, CS195, for participating in the research and adding sunshine to my spring 2002 semester.

Thank you to Dean Tom Storch for supporting the development of the online Higher Order Thinking Skills test for further research at the University of Montana, as well as in other institutions and schools. A thank you is in order to Mike Miller in CIS for providing the dedicated web server to house the online HOTS test.

Of course, big thanks to the University of Montana Computer Science department for providing me with the material and facilities that enabled me to conduct my research. Thank you to Jerry Esmay and Kathy Lockridge for all their help with departmental stuff.

Lastly, thank you Dr. Don Morton for allowing me to pursue my dreams, arranging the offering of an experimental computer science course, CS195, and taking care of all my system administration.

Thank you!

CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

Chapter 1

INTRODUCTION

Society is becoming increasingly more dependent on research and development in the mathematical and computer sciences for technological advances; however, education systems are not producing enough students interested in such jobs. According to the National Science Foundation's science and engineering 2002 indicators, 30% of bachelors and 35% of master's students major in a science and engineering field. However, less than 8% of the US's total workforce is employed in a science and engineering job, and less than 3% actually do science, while the other 5% perform duties such as administration, management, etc. Positively, the 2002 indicators reveal that half of the computer science and math majors work in science and engineering fields (National Science Board of Education, 2002).

Also, the indicators show that many more freshmen have the intent to major in science and engineering disciplines than actually finish. The problem is that few freshmen have the intent to major in computer science and/or mathematics versus all other scientific and engineering disciplines, but jobs for computer specialists are predicted to increase more than any other science and engineering job between the years 2000 – 2010 (National Science Board of Education, 2002). Many CS students have previous degrees, and students return to school to major in computer science to increase their chances of finding a well paying job. The lack of students working in science and engineering fields is a direct result of colleges, high school, and middle school educators either not sustaining interest from students to enter scientific disciplines or improperly preparing students for the next level of learning and reasoning needed to progress through future stages of the sciences. As one author points out, "A third of all eighth-grade ..., while fewer than half of U.S. high-school seniors showed the consistent grasp of fractions, decimals, and percentages expected of competent seventh-graders" (Wilson and Bennett, 1994, p. 5). Dr. Mark Cracolice, from the University of Montana Chemistry department shows evident results when he administered higher order thinking tests to the University of Montana Chemistry freshmen; 75% of the students have insufficient formal thinking skills that should be fully developed by adolescence. Later, this research shows very similar results in introductory computer science classes. Since the Reagan's Nation at Risk document (1983), high school graduates' higher order thinking has remained an issue among educational leaders (p. 8). Formal thinking skills are what enable us to understand and apply computational and logical reasoning that is used in the mathematical sciences, such as computer science.

Table 1-1 Concrete Higher Order Thinking Skills and Description

Types of Thinking Skills	**Brief Description**	**Sample Test Question**
Conservation of volume	Displaced volume is equal to volume of object	Two identical cylinders are filled with the same amount of water. Two metal blocks of the same size and shape are placed side-by-side in the first cylinder. Another two metal blocks are stacked on top of each other and placed into the second cylinder. Which of the following statements is true?
Conservation of liquid amount	Liquid remains the same no matter the size or shape container	A glass is filled with water. Another glass of identical size and shape is filled with the same amount of water as in the first glass. Which of the following is true?
Conservation of length	Length is the same no matter shaper or position	Two lines of identical length are drawn. Arrowheads pointing inwards are placed on line A, while the same arrowheads, but pointing outwards, are placed on line B. Which of the following statements is true?

Table 1-2 Formal Higher Order Thinking Skills and Description

Types of Thinking Skills	**Brief Description**	**Sample Test Question**
Control and exclusion of variables	Holding n independent variables and one dependant variable in mind, and consider effects of each ind. variable	You are provided with a set of pendulums consisting of strings with a number of washers (W) at the end of the string. Suppose you wanted to do an experiment to find out if changing the string length of a pendulum changed the amount of time it takes to swing back and forth. Which pendulums would you use for the experiment?
Ratio and proportion	y=mx (as x goes up, y goes up), and the comparison of two ratios	When a larger gear is turned 8 complete turns, a smaller gear connected to it turned 10 complete turns. If the larger gear is turned 20 complete turns, how many times did the smaller gear turn?
Compensation and equilibrium	yx=m (as y goes up, x goes down), and ab=cd	Tshana was playing with her younger brother on the teeter-totter. Tshana, at 30 kg, sat about 3 feet away from the center of the teeter-totter, as shown in the picture. Where should her younger brother, at 15 kg, sit in order for the teeter-totter to balance?
Correlation	Determination of correlation among variables	Kelley liked to grow flowers. She decided to test a new fertilizer on some of her flowers to see if they would grow larger. At the end of the summer she measured the size of each flower that was treated with fertilizer, and the size of each flower that was not treated with fertilizer. Did the use of fertilizer produce larger flowers?
Probability	Simple sampling procedures; acceptance of the probabilistic nature of natural relationships	A gardener bought a package containing 3 squash seeds and 3 bean seeds. If just one seed is selected from the package what are the chances that it is a bean seed?
Combinatorial	Analyze combinations present in information given	Three students from grade 10, 11, and 12 were elected to student council. A three member committee is to be formed with one person from each grade. All possible combinations must be considered before a decision can be made. Two possible combinations are Tom, Jerry and Dan (TJD), and Sally, Anne, and Martha (SAM). How many combinations are possible (including the two already given)?
Hypothetico-deductive reasoning	Formulate and test alternative hypotheses against given data	A laboratory study is done with white rats. The scientist wants to know if the amount of vitamin A mother rats receive affects the number of offspring born. He uses the same breed of rats in the study. Each rat gets the same amount of food and the same amount of daily exercise. The temperature in all the cages in the same. The hypothesis (explanation) most likely being studied is:

Higher order thinking skills are the skills that allow you to logically reason and can be categorized into two main categories, concrete and formal thinking. These categories are derived from Jean Piaget, who developed a philosophy of cognitive development among children. According to Piaget, an individual should develop their concrete thinking during their later childhood (7-10) and formal thinking occurring during later adolescence around the onset of sexual maturity (11-15), and there are transitional periods between the concrete and formal categories (Richmond, 1970, p. 99). Transitional periods are referred to as low-level and high-level transitions. Within the concrete and formal thinking levels, there are many different types of skills. Some thinking skills associated with concrete thinking are conservation, such as conservation of weight, conservation of length, conservation of area, etc. There are several different types of reasoning skills classified under formal thinking including control and exclusion of variables, proportion, compensation and equilibrium, correlation, probability, combinatorial, and hypothetico-deductive. Some skill types are used more than others in computer science, but an overall formal level of thinking is required. The more common skills used in computer science are combinatorial, hypothetico-deductive, control and exclusion of variables, and correlation reasoning.

Following are examples of questions testing higher order thinking skills. The student is presented with a question and multiple choices to choose from sometimes followed by multiple reasons for the answer. The lowest level of higher order thinking is the concrete level, and an example of a conservation of volume question from the concrete level is "Ryan has sixteen wooden blocks of identical size and shape. He arranges eight of the blocks into the shape of a square. He then arranges the remaining eight wooden blocks into the shape of a rectangle. Which of the following statements is true?" There are several types of skills from the formal thinking level, but a simple example illustrating combinatorial reasoning is "Brian is making a five-course meal for his girlfriend. The courses are appetizer (A), entree (E), main course (M), vegetable (V), and dessert (D). He knows that the appetizer is served first, and the dessert is served last, but he is unsure about what order to use for the entree, main course and vegetable. How many different ways can Brian serve the meal, if he uses each course only once?"

Higher order thinking skills are used throughout computer science, but only a few examples are mentioned in the following paragraph. Control and exclusion of variables type is the thinking required for debugging in computer science because it requires a student to analyze the possible outcomes after the execution of a computer program when he/she changes independent variables in the computer code. Also part of debugging computer code is looking at the relationships between variables in your algorithm, and this is one example of where correlation reasoning is required. Combinatorial reasoning is used when analyzing the bits and bytes needed for numbers and letters in computer architecture. Students need to figure out how many possible combinations there are using a particular number of bits; therefore, flaws, such as overflow, can be detected and understood. Hypothetico-deductive is the most commonly used skill in computer science, because programmers are continuously trying alternative methods of solving a problem when writing a computer program. Hypothetico-deductive reasoning allows the student to formulate and test alternative hypotheses against given data. Therefore, special attention is paid to these types of thinking skills in this research.

This research suggests an astonishing large percentage of University of Montana students do not show the development of formal thinking skills, and this can present a problem to university instructors. College instructors are uneducated in helping to develop these skills, and traditional institutional teaching is designed around rote lecturing, assignments, and tests for those students who have already developed the formal thinking necessary to understand and perform adequately on their own. College instructors need to address this problem and help prepare our students to become future scientists and self-directed learners.

Before one can help correct the problem, one must determine there is a problem among a variety of our students. The research applies to CS195, a beginners' FORTRAN programming class, CS471, an upper level scientific computing class, and CS101, a beginners' Visual Basic programming course, to use as an assessment and evaluation. First in each course, the students are tested for higher order thinking skills. The introduction FORTRAN course is used for the controlled experiment and incorporates new pedagogical methods into the curriculum. Then, the students are retested to assure the instruction facilitates progress. The class is new to the course listings, so there are only 12 students who took the experimental FORTRAN class. The scientific computing and Visual Basic courses are offered yearly and do not use an active learning approach requiring lab work and group discussion. There is always a high enrollment in the Visual Basic course, and there are 46 Visual Basic participants in this research. Even though there is a difference in the number of students enrolled in the experimental versus the traditional introduction course, the pedagogical methods in this research apply to any classroom size with the appropriate planning. This research paper provides a description of the approaches taken to assess the issues in computer science education and help assist in finding an effective way of teaching computer science.

Background

Despite the research conducted on effective ways of educating students at the primary, secondary, and higher education levels, there has not been much recent research in applying new methodologies of teaching to computer science and assessing the effect the new method of teaching has on students' learning. Most articles published in computer journals deal with congruency and material taught in computer science education, and the articles in educational journals analyze and test the problems associated with computer science education. Denning (1989), Schwill (1997), Magel (1989), and Freeman(1997) are computer scientists that have addressed the issues of the fundamental computer science concepts that should be taught in the curriculum, but we try to look at the ways in which the material is presented in the classroom. One present day author, Marcia C. Linn, has researched ways in which computer science and engineering are improved using different approaches in the classroom, but the author never evaluated why the scaffolding framework works better than traditional teaching pedagogical methods (1995). The research in this paper attempts to tackle the problem from both a computer science and educational point of view. When dealing with computer science education, there are many questions addressing educational research. Who is the target audience? What aspects of computer science education, distance learning or classroom environment, is researched? What are the instructional changes, additional tools or new pedagogies, used in the computer science course to enhance students' learning? How is the success of the education research measured? These are questions that have been addressed by computer scientists and educational leaders over the past years, but these questions continue to come up even in present day computer science.

This research only addresses changes in computer science education in a classroom setting. Therefore, no attention was paid to previous research regarding distance learning in computer science education. The research did include pedagogical changes to the instruction of computer science and the inclusion of applications and tools to help relate concepts. There have been several attempts at introducing tools into the computer science curriculum to improve the education through application and interaction as researched by Fung and Pigford. Fung used applications to make the programming concepts easier, and he observed positive results through questionnaires (1996). Whereas, Pigford (2001) and Rodger (1995) introduced computer tools into the classroom to invoke class interaction among the students, and Rodger analyzed an increased awareness among the students in a classroom with an interactive lecture. Both reasons for using computer tools during classroom instruction are valid reasons for improving computer science education. Also, there have been several papers written about the inclusion of active learning and inquiry-based learning into the computer science classroom. Active learning ranges from applications to class discussions to lab work. Jones introduced a participatory classroom in 1987, and he gave many examples of ways to introduce this type of learning into the curriculum. Priebe introduced group interaction in an experimental computer science class and noticed an increase in class attendance among the students (1997).

So, who has been the target audience for computer science classes? Traditionally, computer scientists, mathematicians, engineers, and physicists, who all have strong math and logical thinking skills, have taken computer science. However, we are now seeing students from many other disciplines taking computer science to fulfill core requirements or gain interdisciplinary experience. This new audience proposes a task to introductory computer science educators. There has been previous research in computer science education dealing with teaching programming to non-majors. Biermann investigated teaching to non-computer scientists, and he finds that introducing labs and discussion into the classroom encourages interest among those students (1990). There has been an interest in meeting the needs of both the theoretical and applied computer scientist since 1978 (Jehn, Rine, & Sondak), and this continues to be a problem among computer science departments.

Lastly, how is success measured in computer science education? There has been an assortment of measuring devices including questionnaires, tests, and comparisons with previous students. All of these ways to measure success are valid, but each has its own flaws. We have already discussed some researchers who have used questionnaires and observation (such as class awareness) as ways to measure success, but what are the other possibilities? Ayen from the Air Force Academy used a comparison between present and past students as a comparison, which works well only if the same class is taught by the same professor each semester (1987). However, logical thinking tests are another measurement used to test students' progress through a computer science class. Using the PLT and BCI tests, Wileman found that students with group interaction in the classroom did not increase their logical thinking skills, but they did increase their attendance (1981). More recently, Hudak and Anderson used the FORT logical reasoning test and LSI learning style inventory to assess the student's entering knowledge, but the authors never used the tests as a measuring device for the student's learning or evaluation after teaching (1990). There has not been much research, other than Wileman and Hudak/Anderson, that has used logical thinking tests as a measurement of the success of the computer science education.

The research in this paper is only reflective at the higher education level, but the same techniques can be applied when teaching computer technology at any educational level. Adults and children learn by examples and experiences, and a teacher's job is not to pour information about a subject into students' head so the information can be lost through their ears, but rather facilitates the students' ability to learn and think on their own in the future, when they are presented with problems and challenges within a given subject. The overall aim of education must be to nurture the power of thought (Wees, 1971, p. 59). Computers are the best example of a subject that presents more problems and challenges when nobody is around to help. Computer related subjects are taught best through hands-on experience that exposes students to some likely problems before they become stranded on their own. This is reinforced with Negal's statement, "Application breeds learning" (1994, p. 21).

Many new approaches to teaching, which have all led to increased enrollment and class grades, usually contain application of the concept, group discussions/class work, and hands on experience (National Science Board of Education, 2002). One reason for approaching teaching from many angles comes from the idea that all people learn differently, and therefore teaching needs to be treated as a variable always capable of changing. The way to approach the diversity in learners is with variety of teaching (Draves, 1997, p. 5). Active learning, collaborative learning, inquiry based learning, and discovery learning pedagogies naturally lend themselves to research intensive and experimental disciplines, such as computer science. According to national education interests, recommended science and engineering reforms include a high priority on undergraduate education and research, making faculty aware of new teaching methods, and incorporating interdisciplinary teaching into the curriculum. Institutions currently teach disciplines as they have been taught for years, without incorporating new scientific findings or new methods for the way scientific research is conducted today.

The basic idea is that we need to teach our students for the future. Our institutions currently teach disciplines as they have been taught for years, without incorporating new scientific findings or new methods for the way scientific research is conducted today. As observed by Wees, "For the education of themselves, teachers look ahead. They do it for a reason, for the future. For the education of their children, they look behind." (1971, p. 58). This style of teaching does not help prepare our students for life outside of college, and instead, it hinders ours students' ambitions and competency.

Overview

As mentioned previously, we use three classes, CS195, CS101, and CS471, for our case studies, and we compare students' higher order thinking skills before and after taking each course. These comparisons are used to assess students' thinking at a variety of levels in computer science and using different teaching strategies. Below is an outline and general description of the steps taken in this research to assess the issues in computer science education and help assist in finding an effective way of teaching computer science.

First, a web-based version of the clinical and paper higher-order thinking skills tests is developed to help ease the burden of testing and collecting data for a large number of University of Montana students. The test helps to provide a quick assessment of students' logical thinking skills. Plus, an online version of the test enables an easy way of tracking students' improvement through their college education (see appendix B). We designed the web-based test first so we could continue educational research with relative ease and expand to a larger scale throughout the University of Montana.

Next, we create a control group, CS195. In this control group, we change the teaching methods and left the introductory computer science material the same. Instead of teaching a computer science class using traditional teaching methods, such as only lecturing, CS195 is designed to use updated teaching methods such as active learning, interdisciplinary instruction, and collaborative discussions (see appendix A). The active learning includes both the collaborative discussions and weekly lab experiments. Every week, our instruction relates the computer science topic to other disciplines and application.

Then, we conduct a scientific survey on the effects of the new teaching method. We did this by analyzing the students' thinking skills before and after the course. Students' higher-order thinking skills are retested after taking a class, CS195, which was supposed to promote these skills in the instruction. The overall improvement of students' logical thinking skills are evaluated based on their previous test scores, and the individual thinking skills are evaluated to see the students' exact area of improvement.

Lastly, we compare the current computer science teaching with the new method used in the experiment, CS195, and other pedagogies found to be applicable to computer science. After researching and analyzing different learning theories, each of the learning theories is compared and contrasted to our current teaching strategies at the college level. We accomplish this by sampling a few computer science courses. Also, included in this research is a comparison of the effects our current traditional teaching has on students learning versus the use of more progressive teaching methods.

Chapter 2

CASE STUDY #1

To show there is a need to continue research in computer science education we must first show there is a problem. Then we can propose a solution to the problem, and we can assess and evaluate the solution to the problem. From there, we can continue to conduct research and analyze a worthwhile investigation on our computer science education. For this initial research, a computer science class, CS195, was used as the test set. This course was selected for collecting data, conducting research, and analyzing the results of the research. The computer science course was used to show there is a problem among our college students at all levels, monitor specific teaching methods used for research, and analyze the outcomes after the research.

Selecting the Test Set

To conduct a controlled experiment with valid results, a test set needs to be used that can be controlled and monitored for the specific research. A computer science class, CS195, is created for this research. The class is developed to teach computer programming to scientists of all college levels from many different scientific disciplines, and research on students at various college levels helps show a problem not only at the freshman level but also across all levels, undergraduate and graduate, of college students. The class is oriented for science majors outside of computer science to demonstrate that students, who have not already gained the confidence within the computer science discipline, can understand and do well in computer science if they are presented with computer science topics in such a way that they can relate to and understand. In many entry-level computer science classes, there are students from multiple disciplines, but very rarely is the computer science course taught to meet the needs of many different disciplines. The majority of the computer science classes are taught for the future computer scientist rather than for any scientist, who wants to use computers to write their own scientific programs. The class in this research is designed to teach computer science concepts for use across many different disciplines. The reason for this approach is to show students the inherent interdisciplinary nature of computer science.

Also, we add weekly hands-on lab experience along with weekly class discussions analyzing the many possible solutions to a problem. This is very unique to a computer science course. Both, labs and class discussion, ideas are active learning approaches to computer science education we believe are important in enhancing students' learning. The weekly class discussions are used to solve problems from the previous days lecture through interaction among peers and the instructor. This allows everyone in the classroom (including the instructor) to learn from everyone else's ideas. Next, students are given a lab experiment to conduct alone or with neighboring peers using the instructor's help as needed. This allows the student to find the holes in their understanding of the topic with the teacher present, but we do not currently teach computer science classes in this manner. This class, CS195, is used to test a new approach to computer science education at the college level and analyze the effects by measuring the students' scientific reasoning ability.

Collecting the Data

After specifically selecting and designing CS195 for this research, the students in the course are given a test of scientific reasoning. The test of scientific reasoning tests students for higher order thinking skills, and the test given before the students began CS195 is a revised version of Lawson's test of scientific reasoning skills based on his 1978 article (Lawson, 2000). The students took the test on paper in the beginning of the course, and then at the end of the course, they are given another logical thinking test. The last test given uses the newly developed on-line version that includes questions from various tests and groups: Group Assessment of Logical Thinking (Roadrangka, Yeany, and Padilla, 1983), Group Test of Logical Thinking (Tobin and Capie, 1981), and the University of Montana (Monteyne and Cracolice, 2002). The online version of the clinical tests was developed as a tool for conducting this research (University of Montana, 2002). Not only does this help in future testing, but also this allows many questions from different tests to be combined for more variety. Both tests have a total of fourteen possible points, and the total questions answered correctly out of fourteen categorizes a student's level of thinking: 11-14 (Formal), 7-10 (High Trans), and 3-6 (Low Trans), 0-2 (Concrete). The data is recorded and used for a prediction and assessment of the students' performance in the class. Not only is the data from the tests used to assess the students' performance, but also the data is used to show the effect the new teaching method had on the development of the students' thinking skills.

Below is a table of the information collected from the students in CS195. On the first day of class, all 12 students took the test as a course assignment in a monitored examination setting. On the last day of class, students took the test along with traditional instructor evaluation on the last day of class before the final examination, without formal proctoring, and 7 students attended class on that day, as compared with 12 that took the test in the beginning. The table consists of the students' ages, class status, and test scores before and after the course, as well as their overall performance in CS195. Due to the confidentiality of student information, there are no names or social security numbers associated with the student data in Table 2-1, or any of the tables. Some data is not available because of either insufficient information on the student's test form or incompletion of the CS195 course resulting from a dropped or failed status.

Table 2-1 CS195 Student Info & HOTS Test Scores

Student	**Age**	**Class Status**	**1st Test Score (14 total)**	**2nd Test Score (14 total)**	**CS195 Grade**
1	19	Freshman	7	N/A	dropped
2	42	Other	8	N/A	dropped
3	N/A	Junior	9	N/A	dropped
4	27	Freshman	7	N/A	failed
5	24	Senior	8	9	C
6	23	Senior	10	10	A
7	23	Senior	13	12	A
8	21	Freshman	12	14	B
9	24	Freshman	14	14	A
10	29	Other	10	14	A
11	29	Other	10	12	A
12	22	Senior	9	N/A	failed

The next two tables (Table 2-2, 2-3) of data shown are the test results of individual higher order thinking skills for each of the students taking the test before and after CS195. Combinatorial reasoning is not shown in either table because neither the Lawson nor the on-line test (not at that time) contained any combinatorial questions in the database. The first table does not include the compensation and equilibrium skill because the Lawson test given before CS195 did not include this skill. When many tests were combined, the compensation and equilibrium skill is represented in the database of questions for the on-line version. The Lawson test given first has a preset number of questions for each skill, and all students are tested on the same skill types and the same number of questions for each skill type. The number of questions presented from a skill type is represented in parenthesis beside the thinking skill, and the numbers in the table correspond to a student and the number of questions the student answered correctly from the skill type. Refer to Table 1-1 and Table 1-2 for a brief description of each of the various thinking skills.

Table 2-2 Before CS195 Higher Order Thinking Skills

Student	Prob (4)	Consv (2)	Control (1)	Prop (2)	Corr (1)	Hypo (4)	Total (14)
1	2	1	1	0	1	2	7
2	2	2	0	2	0	2	8
3	3	2	1	1	1	1	9
4	4	1	1	1	0	0	7
5	2	2	1	1	1	1	8
6	2	2	1	2	1	2	10
7	3	2	1	2	1	4	13
8	4	2	1	1	0	4	12
9	4	2	1	2	1	4	14
10	3	2	1	1	0	3	10
11	2	2	1	2	0	3	10
12	2	2	1	1	0	3	9

At the time, the on-line version of the higher order thinking skills test chose ten questions from different thinking skills randomly but always starting with a conservation type question and ending with four hypothetico-deductive reasoning questions. So, the student results from the on-line test given after CS195 show each student's results containing a different number of thinking skills and a different number of questions for each skill tested. The number in parenthesis contains the total number of questions of that skill presented to each student and the other number represents the total questions answered correctly out of those questions presented.

Table 2-3 After CS195 Higher Order Thinking Skills

Student	Prob	Consv	Control	Prop	Corr	Comp	Hypo (4)	Total (14)
5	0 (1)	2 (2)	0 (1)	3 (3)	0 (1)	2 (2)	2	9
6	1 (1)	2 (2)	2 (2)	2 (2)	1 (1)	1 (2)	1	10
7	1 (3)	3 (3)	1 (1)	2 (2)	1 (1)	0 (0)	4	12
8	3 (3)	3 (3)	1 (1)	2 (2)	0 (0)	1 (1)	4	14
9	2 (2)	1 (1)	3 (3)	3 (3)	1 (1)	0 (0)	4	14
10	2 (2)	2 (2)	2 (2)	3 (3)	1 (1)	0 (0)	4	14
11	1 (2)	1 (1)	2 (2)	2 (3)	1 (1)	0 (0)	4	12

Conducting the Research

To find out if there is an effective way to teach computer science other than using traditional methods, new teaching practices are implemented into the CS195 FORTRAN classroom. Three new additions are made to the classroom instruction. One addition to the computer science course is the addition of hands-on experience through a lab experiment conducted each week in the classroom. Another addition to the course is the weekly group discussions. The group/class discussion uses the whole class to solve a problem. The problems proposed in this class are unique because they are tailored to meet the needs of students' scientific backgrounds. The CS class is taught using an interdisciplinary approach combining math, CS, and other sciences revealing the interdisciplinary nature of computer science.

Within most scientific disciplines, there is a lab associated with the course, which introduces students to the concepts covered in class through hands-on experience. Currently, computer science does not have a lab with the courses, yet computer science is as experimental and hands-on as any other scientific discipline. Many computer science concepts cannot become concrete unless students are given the hands-on experiences that make the concepts clearer. Also, students learn through their own experiences. Labs in computer science help students overcome the obstacles of the computer science terminology and hardships. The labs allow the students to find places where their knowledge is missing or misconstrued, and they are a way for the students to have fun and stimulate their interests in computer science before they become frustrated and give up.

Another approach to active learning in the classroom is group discussion and interaction among other students and the teacher. Professors are there to help the students become better people through increasing their ability to think for themselves in a discipline. According to Greta Negal, "Do not hesitate to express your interest in getting to know others or to admit that you have much to learn from them" (1994, p. 179). The group discussions not only break communication barriers between the students and teacher but also can help a teacher assess what the students do not correctly understand. Another important issue conveyed through group discussion is the fact that there is more than one way to do the same thing on a computer. Therefore, group discussion to a solution can permit students to see opposing viewpoints on issues (Negal, 1994, p. 121).

The last addition to the FORTRAN class is the application of programming concepts to real world problems. Wees (1971) states that "Force breeds fear" (p. 16). Therefore, applying CS to already known interests helps to stimulate rather than suppress the student's excitement. To many students, computer science is already foreign and the concepts in computer science are very abstract until they are applied to a problem of interest to the student. Application is accomplished in the lecture and group discussion, but it can also be put into projects that the students are able to create based on their interests. Negal suggests asking the children what they would like to do and guide them into ways of accomplishing those things with an eye to further learning (1994, p. 95). This helps a student become interested in a topic and increases the student's understanding and ability to progress in the subject on their own. The adult's interest in solving problems within their older time perspective makes adults more concerned with specific, narrow topics of relevance than broad, generalized or abstract subjects (Draves, 1997, p. 9). Computer science is inherently interdisciplinary, but it is also very theoretical and mathematical. We do not want to confuse this research with pure, theoretical computer science education, which is intended for those students wanting to study the raw science of computers and how the operate. We are discussing computer science education for those students looking for the application.

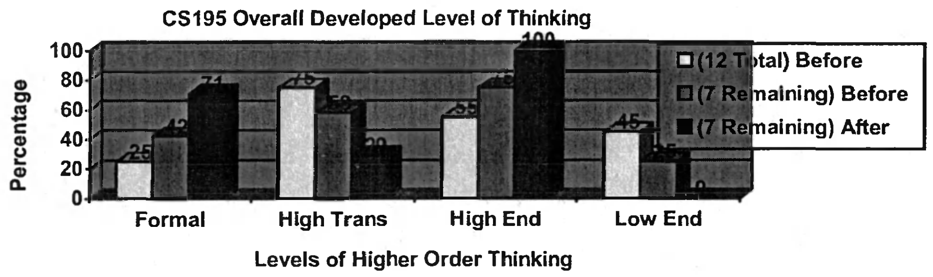
Analysis of the Data

The data collected in this research is used to assess the students' knowledge, predict the students' ability to handle a course requiring formal thinking skills, and assess the students' improvement in thinking ability under a controlled teaching environment. Data from the CS195 students' higher order thinking skills tests is collected in the beginning of the FORTRAN course to determine the level of thinking developed among a diverse group of individuals, who do not have any previous instruction in computer science. After teaching FORTRAN by applying new techniques for teaching, the students' higher order thinking skills are retested to see if there is any improvement in their thinking skills after taking the class. Also, each skill that is tested is analyzed to see the exact type of thinking skills that increased among the remaining CS195 students. Not only does this test provide data about student's developed thinking skills, but the test results serve as a prediction about how well a student might perform in a computer science class that requires certain formal thinking skills. The following percentages for different levels are taken from the scoring of the Lawson test. The percentages are as follows: 78.6-100% (Formal), 50-71.5% (High Trans), and 21.4-42.9% (Low Trans), 0-14.3% (Concrete).

The data collected from the HOTS test before starting the experimental FORTRAN class provides a good indication of the overall level of thinking among the students. Most of the students in the class have not developed their formal thinking skills, but none of the students fell below the high transitional level. There are about an equal amount of students that fall into the higher end versus the lower end of the high-level transition stage. This is as expected because the course is promoted to students that already have an interest in science. The majority of our college students need to be at the formal thinking level, but if they are not at the formal thinking level, then one hopes they are in the higher end of the high transitional period.

Because there are twelve students that took the test before CS195 and seven that retested after CS195, we cannot compare these two percentages other than analyzing possible causes for fluctuation in the class's overall level of thinking. Reviewing the class's initial versus ending level of thinking offers some clues about the level of thinking required to stay and pass a computer science course requiring formal thinking skills. The CS195 class began with an overall lower level of skills than the level of skills in the end of the course. This is a result of students with a lower level of thinking dropping/failing the course or an increase in the remaining students' level of thinking.

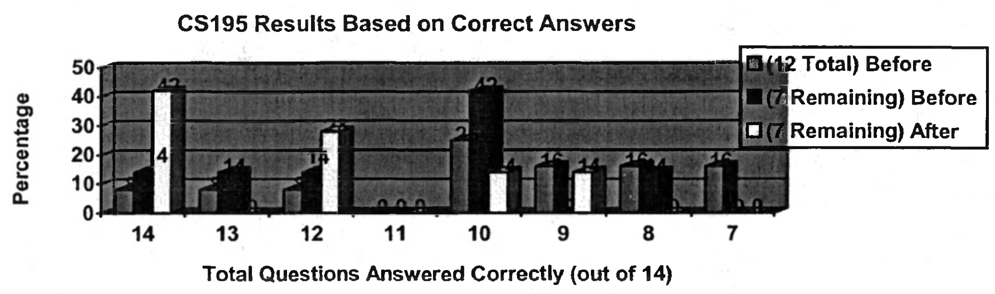
Figure 1



To gain a better understanding of the effects the alternative teaching have on the students, only the students that took both tests are compared to get accurate percentages for the levels of thinking skills before and after either class. The data from the remaining students before and after FORTRAN shows that there is indeed a 29% increase in the class's formal level of thinking, as well as a movement of developed skills from the lower end to the higher end of the high transitional. This shows that either the teaching is effective on developing thinking skills or taking a computer science course stimulates a student's higher order thinking that is previously suppressed. In either case, the results show a positive outcome for the experimental course. The traditional Visual Basic course did not show any gain in the remaining students' level of reasoning skills. In fact, it looks as if there is a decline in the overall class's high transitional level and an increase in percentage in the low transition level.

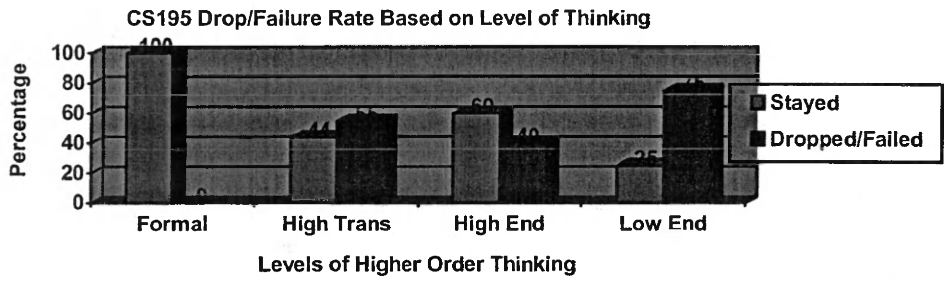
Another assessment of the class's ability is to look at the total questions answered correctly in the beginning and end of the course, and comparing both the initial and the remaining students' test scores before the course with the test results from the remaining students after the course. This helps to gauge the size of improvement in terms of total questions answered correctly vs. the level of thinking. According to Figure 2 results, there is a large decrease in 10 questions and a large increase in 14 questions, the total possible questions, answered correctly. This is a positive improvement, and it shows an upward trend in the class's level of thinking. The gap between 10-14 questions answered correctly is an enormous improvement in a student's level of thinking.

Figure 2



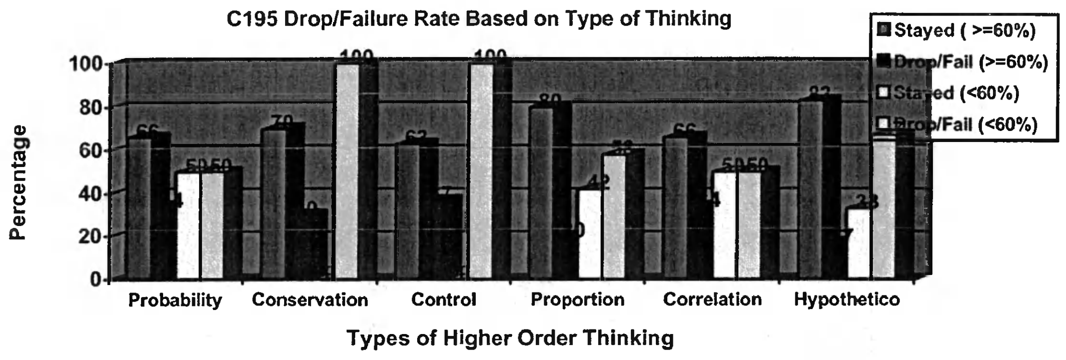
Next, did the students' level of thinking or types of thinking skills influence either their decision to continue with the class or the grade they receive in the end of the class? According to the results shown below, students with formal thinking skills are more inclined to stay and pass a course requiring logical thinking versus students without formal thinking. Also, one can see that the closer students are to the formal thinking level, the higher the chances are of success.

Figure 3



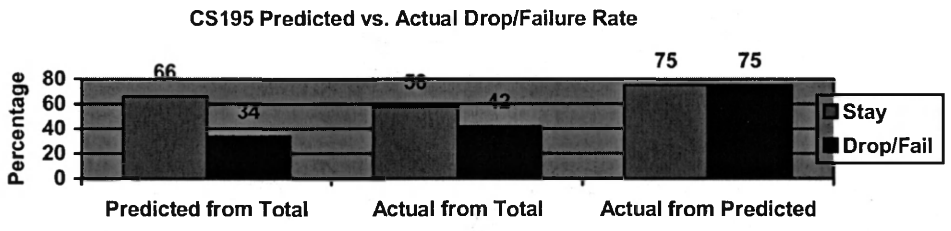
Even though Figure 3 results reveal a high correlation between a student's level of thinking and their ability to succeed, the figure below does not show such a distinct correlation between the types of thinking skills developed and the student's ability. Also, the results below for the experimental course, CS195, show that students without conservation and control of variables skills are 100% likely to either drop or fail a computer science class, and a majority of the students that lack ratio and proportion and hypothetico-deductive reasoning skills have an increased likelihood to drop/fail. Students have an increased likelihood to drop if they do not have conservation skills, but there is not an indication that students without certain skills will drop/fail. Refer to Table 1-1 and Table 1-2 for more details on the different thinking skills. It makes sense that students without conservation skills, which make up the concrete thinking level, might not be able to handle a course requiring a much higher level of thinking. This information can be very useful to advisors when they are helping to place students in classes.

Figure 4



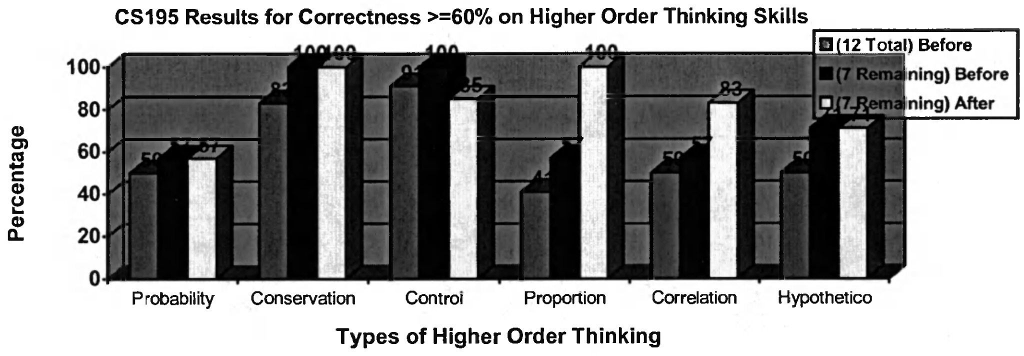
The data below in Figure 5 shows the actual drop/failure rate versus the predicted drop/failure rate for CS195. The predicted value is based on the student's level of thinking, and if a student has reached the formal level of thinking or fell within the higher end of the high-level transition (9-14 or ≥60%), then he/she is predicted to have the skills needed to stay in the class. As you can see from the figures below, the predicted percentage of students that would either stay or drop/fail is very close to the actual percentage of students, who did stay or drop/fail. Even though these results give us an indication of the overall class performance based on the student's level of thinking, this does not give us an idea about whether the particular students predicted to stay or drop/fail actually did so. The last set of results show the actual percentage of students that either stayed or dropped/failed out of the predicted students to stay or drop/fail. The actual percentage of students that either stayed or dropped/failed from those predicted is 75-80%, and this shows a strong correlation between students that have a higher versus a lower level of thinking.

Figure 5



Previously from Figures 1 and 2, we see an overall improvement in the remaining students' level of thinking and test scores. We have not analyzed the actual skills needed for a computer science course and increase in students' skill types because of CS195. Use Table 1-1 and Table 1-2 for more details about the higher order thinking skills. According to the figure below, conservation and control are the two skills over half the class have developed before entering the experimental course, and the types of skills developed by the remaining students before and after CS195 are not much of an improvement, except in the proportion and correlation skills. This shows that not any one type of skill hinders the student's performance in such a class, but their overall level of thinking has much more of an effect. In the skill used most often, hypothetico-deductive reasoning, there is the greatest distance between the number of students that had the skill developed from the initial versus the remaining FORTRAN students. This may show that this skill is needed more to succeed in a computer science course. The second skill that is developed by more of the remaining students is conservation, which is the first higher order thinking skill learned and part of the concrete level of thinking developed before formal thinking. From the remaining students in CS195, proportion and correlation reasoning are the only two skills that increased after the class.

Figure 6


CS195 Results for Correctness >=60% on Higher Order Thinking Skills
(12 Total) Before
(7 Remaining) Before
(7 Remaining) After
Percentage
100
80
60
40
20
0
100
100
100
85
100
83
Probability
Conservation
Control
Proportion
Correlation
Hypothetico
Types of Higher Order Thinking

Chapter 3

CASE STUDY #2

After data was collected from a computer science course, which used progressive teaching strategies, this data is compared to other computer science classes from different levels that used other methods of teaching. An upper level computer science course, CS471, is chosen to compare the skills held by students already attaining a computer science degree versus those just beginning in the computer science field. Also, CS471 uses a more progressive way of teaching computer science by relating scientific computing to the atmospheric sciences. The other computer science class selected for analysis is a beginners Visual Basic course, which is for the non-computer scientists. This course, CS101, has a wide variety of students varying in levels and disciplines. Choosing both computer science courses, CS471 and CS101, to compare against our controlled experiment helps to give insight about students' capabilities at different levels and from different fields as well as the effects different teaching methods have on students learning.

Selecting the Test Set

The test sets were chosen carefully for this case study, because we want to maximize the differences between the two case studies being compared. The first case study is our controlled experiment using progressive teaching strategies in a classroom with mixed majors and levels, and the classes used in this case study for comparison are not controlled and more traditional. All of the classes used in both case studies contained a variety of students from different disciplines. The second case study contained one class that has students with interests in the sciences and computer science while the other course included students that may or may not have any interests in the sciences, much less computer science. These test sets are chosen for the second case study to analyze the differences between students with interests in the scientific disciplines from case study one with upper level students interested in computer science and students without any interests in the sciences. Comparing and analyzing students from these scenarios gives us a better understanding of the abilities required to do computer science and the effects our teaching has on students with and without logical reasoning needed to understand and perform well in a computer science class. The professors from the classes chosen for the second case study did not use the same active learning techniques, lab work and group discussion, used in the experimental CS195 course, and therefore, the classes more closely represent the current traditional teaching strategies used in undergraduate computer science classes.

Collecting the Data

The students from both test sets in the second case study are given the online version of the higher order thinking skills tests. The online test is a combination of several scientific reasoning tests combined. All of the higher order thinking skills types are tested except hypothetico-deductive reasoning. We did not test for this skill because we did not have enough questions for the current design of the online test. The current online version of the HOTS test presents a student with a medium level question from a random skill, and if the student answers the question correctly, then he/she is presented a hard level question of the same skill type (see appendix B). The student completes a skill type when he/she answers two of any level (easy, medium, or hard) correctly without previously failing two questions in the same level, and if the student fails to answer two of all the levels correctly, then he/she is not rated at any level for the skill. The higher order thinking skills test is given twice to the CS471 and CS101 classes to analyze the thinking skills held by upper and lower level computer science students and how the students' skills are influenced by the computer science class and instruction.

There are two tables below showing test scores and other information for both CS471 and CS101 classes respectively. On the first day of class, all 46 students in CS101 and 4 students in CS471 took the test as a course assignment. On the last day of class, students in CS101 took the test along with traditional instructor evaluation on the last day of class before the final examination, without formal proctoring, and 34 students attended class on that day, as compared with 46 that took the test in the beginning. Table 3-1 and 3-2 show the student's age, test scores before and after taking the computer science, and their final grade in the class, but the students' names are not revealed at any time for confidentiality. The number of questions answered correctly precedes the total number of questions presented to the student that is shown in parenthesis.

Table 3-1 CS471 Student Info & HOTS Test Scores

Student	Age	Class Status	1st Test Score	2nd Test Score	CS471 Grade
1	24	Other	12 (15)	N/A	N/A
2	22	Senior	16 (17)	N/A	N/A
3	41	Other	21 (27)	N/A	N/A
4	29	Other	17 (21)	N/A	N/A

Table 3-2 CS101 Student Info & HOTS Test Scores

Student	Age	Class Status	1st Test Score	2nd Test Score	CS101 Grade
1	23	Junior	16 (24) – 66.7	14 (30) – 46.7	98
2*	21	Freshman	4 (21) – 19.0	N/A	51
3	26	Freshman	19 (26) – 73.1	20 (22) – 90.9	95
4	25	Freshman	17 (25) – 68.0	16 (25) – 64.0	93
5	29	Freshman	13 (29) – 44.8	N/A	26
6	26	Sophmore	17 (26) – 65.3	18 (24) – 75.0	77
7*	19	Freshman	2 (19) – 10.5	0 (18) – 0.0	Dropped
8	26	Freshman	10 (26) – 38.5	17 (27) – 63.0	89
9*	25	Freshman	19 (25) – 76.0	5 (25) – 20.0	95
10*	31	Senior	11 (31) – 35.5	4 (20) – 20.0	75
11	29	Senior	18 (29) – 62.1	19 (26) – 73.1	99
12	19	Freshman	17 (19) – 89.5	19 (20) – 95.0	97
13	33	Freshman	20 (33) – 60.1	17 (23) – 73.9	99
14	33	Senior	18 (33) – 54.5	10 (25) – 40.0	94
15	29	Freshman	18 (29) – 62.0	22 (35) – 62.9	96
16	27	Freshman	10 (27) – 37.0	N/A	23
17	27	Sophmore	18 (27) – 66.7	N/A	dropped
18	27	Junior	21 (27) – 77.8	N/A	63
19	27	Freshman	18 (27) – 66.7	16 (26) – 61.5	97
20	24	Freshman	15 (24) – 62.5	20 (30) – 66.7	87
21*	30	Freshman	10 (30) – 33.3	N/A	17
22	32	Freshman	24 (32) – 75.0	21 (28) – 75.0	99
23*	26	Freshman	5 (26) – 19.2	8 (22) – 36.4	94
24	21	Sophmore	17 (21) – 81.0	17 (22) – 77.3	99
25	26	Sophmore	15 (26) – 57.7	14 (27) – 51.9	99
26*	27	Sophmore	14 (27) – 51.9	12 (33) – 36.4	76
27	24	Freshman	18 (24) – 75.0	N/A	92
28*	21	Freshman	2 (21) – 9.5	3 (24) – 12.5	89
29*	33	Senior	12 (33) – 36.4	0 (18) – 0.0	93
30	24	Sophmore	20 (24) – 83.3	N/A	96
31	24	Freshman	15 (24) – 62.5	13 (26) – 50.0	60
32	26	Sophmore	15 (26) – 57.7	14 (24) – 58.3	dropped
33	29	Sophmore	17 (29) – 58.6	13 (29) – 44.8	91
34	27	Junior	18 (27) – 66.7	17 (28) – 60.7	97
35*	24	Freshman	11 (24) – 45.8	N/A	dropped
36*	20	Sophmore	1 (20) – 5.0	3 (24) – 12.5	89
37	30	Freshman	17 (30) – 56.7	22 (31) – 71.0	96
38*	31	Junior	17 (31) – 54.8	4 (24) – 16.7	92
39*	23	Junior	6 (23) – 26.1	2 (22) – 9.1	82

40	28	Freshman	21 (28) – 75.0	19 (27) – 70.4	81
41*	27	Freshman	6 (27) – 22.2	N/A	30
42*	21	Senior	3 (21) – 14.3	N/A	dropped
43	27	Senior	19 (27) – 70.4	18 (28) – 64.3	95
44	29	Freshman	15 (29) – 51.7	N/A	89
45	25	Freshman	10 (25) – 40.0	12 (31) – 38.7	68
46	31	Freshman	16 (31) – 51.6	N/A	56
47	24	Other	19 (24) – 79.2	21 (27) – 77.8	98
48	36	Sophmore	21 (36) – 58.3	N/A	89
49	25	Freshman	20 (25) – 80.0	N/A	23
50	27	Freshman	18 (27) – 66.7	N/A	dropped
51	26	Senior	21 (26) – 80.8	17 (24) – 70.8	95
52	33	Freshman	22 (33) – 66.7	N/A	34
53	30	Freshman	21 (30) – 70.0	11 (26) – 42.3	99
54	28	Senior	15 (28) – 53.6	9 (25) – 36.0	94
55	26	Freshman	13 (26) – 50.0	17 (28) – 60.7	96
56	24	Freshman	11 (24) – 45.8	20 (24) – 83.3	88
57	28	Freshman	19 (28) – 67.9	18 (29) – 62.1	96
58	26	Senior	20 (31) – 64.5	15 (27) – 55.6	99
59	31	Sophmore	20 (26) – 76.9	14 (22) – 63.6	99
60	33	Other	21 (33) – 63.6	19 (32) – 59.4	77
61	27	Sophmore	15 (26) – 57.7	11 (23) – 47.8	88

* - Faulty information due to invalid entries or invalid test completion time

The next two tables (Table 3-3, 3-4) of data shown are the test results of individual higher order thinking skills for each of the students taking the test before CS471 and CS101. Hypothetico-deductive reasoning is not shown in either table because the current version of the online test used did not include testing this skill at the time of this research. There is a minimum total number of questions the student may have presented to himself/herself (3 per skill), and some of the students that took the online test very early in the CS471 class were only presented with five different types of skills, i.e. total >= 15. However, the rest of the students in CS471 and all students from CS101 are tested on six higher order thinking skill types. The total questions presented to each student are represented in parenthesis, and the other number outside parenthesis is the number of questions answered correctly by the student. Because some students used an older version of the on-line test in the beginning of the semester, student answers for a skill type may not be available. Please make sure to refer to Table 1-1 and Table 1-2 for more details regarding the different higher order thinking skills tested in the second case study.

Table 3-3 Before CS471 Higher Order Thinking Skills

Student	Prob	Consv	Control	Prop	Corr	Comb	Total (>=15)
1	3 (3)	3 (3)	0 (3)	3 (3)	3 (3)	N/A	12 (15)
2	3 (3)	3 (3)	4 (5)	3 (3)	3 (3)	N/A	16 (17)
3	4 (5)	4 (5)	3 (3)	3 (3)	3 (6)	4 (5)	21 (27)
4	3 (3)	3 (5)	3 (3)	3 (3)	3 (3)	2 (4)	17 (21)

Table 3-4 Before CS101 Higher Order Thinking Skills

Student	Prob	Consv	Control	Prop	Corr	Comb	Total (>=18)
1	3 (5)	3 (3)	3 (5)	3 (3)	0 (3)	4 (5)	16 (24)
2*	2 (4)	0 (3)	0 (3)	0 (3)	0 (3)	2 (5)	4 (21)
3	3 (3)	4 (5)	3 (3)	2 (4)	3 (6)	4 (5)	19 (26)
4	3 (3)	3 (3)	3 (3)	3 (6)	2 (5)	3 (5)	17 (25)
5	3 (6)	3 (3)	0 (3)	2 (5)	1 (5)	4 (7)	13 (29)
6	3 (5)	3 (3)	3 (3)	2 (4)	3 (6)	3 (5)	17 (26)
7*	2 (4)	0 (3)	0 (3)	0 (3)	0 (3)	0 (3)	2 (19)
8	3 (5)	0 (3)	0 (3)	3 (6)	2 (5)	2 (4)	10 (26)
9*	3 (3)	4 (5)	4 (5)	3 (3)	2 (4)	3 (5)	19 (25)
10*	4 (7)	0 (3)	1 (5)	2 (4)	3 (7)	1 (5)	11 (31)
11	4 (5)	3 (3)	2 (4)	4 (7)	2 (7)	3 (3)	18 (29)
12	3 (3)	3 (3)	3 (3)	3 (3)	2 (4)	3 (3)	17 (19)
13	2 (4)	3 (3)	5 (8)	4 (5)	2 (6)	4 (7)	20 (33)
14	4 (5)	3 (3)	3 (8)	2 (4)	2 (7)	4 (6)	18 (33)
15	3 (3)	3 (3)	2 (4)	4 (8)	2 (4)	4 (7)	18 (29)
16	2 (4)	3 (6)	2 (5)	0 (3)	0 (3)	3 (6)	10 (27)
17	5 (7)	3 (3)	2 (4)	3 (3)	3 (6)	2 (4)	18 (27)
18	3 (3)	3 (3)	4 (6)	3 (3)	4 (5)	4 (7)	21 (27)
19	4 (8)	3 (3)	3 (3)	3 (3)	2 (4)	3 (6)	18 (27)
20	3 (3)	3 (3)	2 (4)	3 (3)	1 (5)	3 (6)	15 (24)

21*	3 (6)	2 (4)	2 (7)	2 (5)	1 (5)	0 (3)	10 (30)
22	3 (3)	4 (5)	5 (7)	5 (7)	4 (5)	3 (5)	24 (32)
23*	0 (3)	0 (3)	1 (5)	2 (5)	0 (3)	2 (7)	5 (26)
24	3 (3)	3 (3)	3 (3)	3 (3)	2 (4)	3 (5)	17 (21)
25	3 (3)	3 (3)	0 (3)	4 (7)	1 (5)	4 (5)	15 (26)
26*	3 (3)	4 (5)	1 (5)	3 (5)	0 (3)	3 (6)	14 (27)
27	2 (4)	3 (3)	2 (4)	4 (5)	3 (3)	4 (5)	18 (24)
28*	2 (6)	0 (3)	0 (3)	0 (3)	0 (3)	0 (3)	2 (21)
29*	4 (7)	1 (5)	3 (7)	1 (5)	2 (4)	1 (5)	12 (33)
30	4 (5)	3 (3)	4 (5)	3 (3)	3 (3)	3 (5)	20 (24)
31	4 (5)	3 (3)	0 (3)	3 (5)	2 (5)	3 (3)	15 (24)
32	4 (6)	3 (3)	3 (3)	2 (4)	0 (3)	3 (7)	15 (26)
33	3 (5)	3 (3)	3 (7)	3 (3)	2 (4)	3 (7)	17 (29)
34	2 (4)	4 (5)	4 (5)	2 (4)	2 (4)	4 (5)	18 (27)
35*	4 (5)	3 (3)	0 (3)	3 (5)	0 (3)	1 (5)	11 (24)
36*	0 (3)	0 (3)	0 (3)	0 (3)	0 (3)	1 (5)	1 (20)
37	3 (5)	3 (3)	5 (7)	3 (3)	2 (7)	1 (5)	17 (30)
38*	4 (5)	3 (5)	4 (5)	3 (8)	0 (3)	3 (5)	17 (31)
39*	3 (5)	0 (3)	0 (3)	0 (3)	2 (4)	1 (5)	6 (23)
40	4 (5)	3 (3)	3 (3)	4 (5)	4 (7)	3 (5)	21 (28)
41*	2 (7)	2 (4)	0 (3)	0 (3)	1 (5)	1 (5)	6 (27)
42*	0 (3)	0 (3)	0 (3)	3 (6)	0 (3)	0 (3)	3 (21)
43	3 (5)	3 (3)	4 (5)	4 (5)	2 (4)	3 (5)	19 (27)
44	2 (4)	4 (5)	3 (3)	3 (5)	1 (5)	2 (7)	15 (29)
45	2 (4)	3 (3)	1 (5)	2 (5)	0 (3)	2 (5)	10 (25)
46	2 (4)	5 (7)	4 (7)	2 (4)	2 (4)	1 (5)	16 (31)
47	3 (3)	3 (3)	3 (5)	3 (3)	3 (3)	4 (7)	19 (24)
48	3 (6)	3 (3)	3 (8)	4 (5)	4 (7)	4 (7)	21 (36)
49	3 (3)	3 (3)	3 (3)	4 (5)	4 (8)	3 (3)	20 (25)
50	3 (6)	3 (3)	3 (3)	2 (4)	4 (5)	3 (6)	18 (27)
51	3 (3)	3 (3)	3 (3)	4 (5)	4 (5)	4 (7)	21 (26)
52	3 (5)	3 (3)	5 (7)	4 (5)	3 (6)	4 (7)	22 (33)
53	4 (7)	4 (5)	4 (5)	3 (5)	3 (3)	3 (5)	21 (30)
54	5 (7)	3 (3)	0 (3)	0 (3)	2 (5)	5 (7)	15 (28)
55	2 (4)	3 (3)	2 (7)	3 (3)	0 (3)	3 (6)	13 (26)
56	1 (5)	3 (3)	2 (4)	2 (4)	0 (3)	3 (5)	11 (24)
57	3 (3)	3 (3)	4 (5)	3 (6)	3 (5)	3 (6)	19 (28)
58	5 (7)	3 (3)	2 (5)	4 (7)	2 (4)	4 (5)	20 (31)
59	3 (3)	4 (5)	3 (3)	3 (3)	3 (7)	4 (5)	20 (26)
60	4 (7)	3 (3)	5 (7)	3 (6)	3 (5)	3 (5)	21 (33)
61	3 (3)	3 (3)	3 (6)	2 (4)	0 (3)	4 (7)	15 (26)

* - Faulty information due to invalid entries or invalid test completion time

Table 3-5 shows the student's results for individual higher order thinking skills after taking CS101. We did not retest CS471 because their skills were so high before taking the CS471 course, and we do not suspect students' level of thinking will regress.

Table 3-5 After CS101 Higher Order Thinking Skills

Student	Prob	Consv	Control	Prop	Corr	Comb	Total (>=18)
1	2 (4)	4 (5)	3 (6)	1 (5)	0 (3)	4 (7)	14 (30)
3	3 (3)	4 (5)	3 (3)	3 (3)	4 (5)	3 (3)	20 (22)
4	2 (4)	3 (3)	3 (5)	4 (5)	2 (4)	2 (4)	16 (25)
6	3 (3)	3 (3)	3 (3)	4 (6)	3 (5)	2 (4)	18 (24)
7*	0 (3)	0 (3)	0 (3)	0 (3)	0 (3)	0 (3)	0 (18)
8	4 (5)	3 (3)	3 (5)	3 (6)	2 (4)	2 (4)	17 (27)
9*	1 (5)	1 (5)	0 (3)	0 (3)	0 (3)	3 (6)	5 (25)
10*	0 (3)	0 (3)	0 (3)	0 (3)	0 (3)	4 (5)	4 (20)
11	2 (4)	3 (3)	3 (3)	4 (5)	3 (6)	4 (5)	19 (26)
12	3 (3)	3 (3)	3 (3)	3 (3)	3 (3)	4 (5)	19 (20)
13	2 (4)	3 (3)	3 (3)	4 (5)	3 (3)	2 (5)	17 (23)
14	3 (5)	3 (3)	1 (5)	2 (4)	0 (3)	1 (5)	10 (25)
15	5 (7)	4 (5)	4 (5)	3 (6)	1 (5)	5 (7)	22 (35)
19	4 (5)	3 (3)	4 (6)	2 (4)	0 (3)	3 (5)	16 (26)
20	2 (4)	4 (5)	4 (5)	4 (5)	2 (4)	4 (7)	20 (30)
22	3 (3)	3 (3)	4 (5)	4 (5)	4 (7)	3 (5)	21 (28)
23*	1 (5)	3 (3)	0 (3)	2 (4)	0 (3)	2 (4)	8 (22)
24	3 (3)	3 (3)	3 (3)	3 (3)	2 (5)	3 (5)	17 (22)
25	3 (5)	4 (5)	1 (5)	4 (5)	0 (3)	2 (4)	14 (27)
26*	2 (4)	4 (5)	1 (5)	1 (5)	2 (7)	2 (7)	12 (33)
28*	0 (3)	0 (3)	0 (3)	1 (5)	1 (5)	1 (5)	3 (24)
29*	0 (3)	0 (3)	0 (3)	0 (3)	0 (3)	0 (3)	0 (18)
31	3 (5)	3 (3)	4 (7)	3 (5)	0 (3)	0 (3)	13 (26)
32	4 (5)	3 (3)	3 (5)	0 (3)	2 (4)	2 (4)	14 (24)
33	2 (4)	2 (7)	0 (3)	3 (5)	3 (7)	3 (3)	13 (29)
34	3 (3)	3 (3)	3 (3)	3 (7)	1 (5)	4 (7)	17 (28)
36*	0 (3)	0 (3)	0 (3)	1 (5)	1 (5)	1 (5)	3 (24)
37	5 (7)	3 (3)	4 (5)	3 (5)	3 (6)	4 (5)	22 (31)
38*	0 (3)	0 (3)	1 (5)	2 (5)	1 (5)	0 (3)	4 (24)
39*	0 (3)	1 (5)	0 (3)	1 (5)	0 (3)	0 (3)	2 (22)
40	4 (5)	4 (5)	3 (3)	3 (5)	2 (4)	3 (5)	19 (27)
43	3 (5)	3 (3)	3 (3)	3 (3)	3 (7)	3 (7)	18 (28)

45	2 (4)	0 (3)	5 (8)	4 (8)	0 (3)	1 (5)	12 (31)
47	4 (5)	3 (3)	3 (3)	4 (5)	3 (6)	4 (5)	21 (27)
51	3 (5)	3 (3)	3 (3)	3 (3)	3 (6)	2 (4)	17 (24)
53	2 (5)	5 (7)	1 (5)	0 (3)	0 (3)	3 (3)	11 (26)
54	3 (5)	3 (3)	0 (3)	0 (3)	3 (8)	0 (3)	9 (25)
55	2 (4)	3 (3)	2 (4)	5 (7)	3 (6)	2 (4)	17 (28)
56	4 (5)	3 (3)	3 (3)	3 (3)	3 (5)	4 (5)	20 (24)
57	3 (6)	3 (3)	3 (6)	3 (5)	2 (4)	4 (5)	18 (29)
58	2 (4)	3 (3)	2 (4)	3 (6)	3 (6)	2 (4)	15 (27)
59	3 (3)	3 (3)	2 (4)	3 (3)	1 (5)	2 (4)	14 (22)
60	3 (3)	3 (5)	4 (5)	3 (3)	2 (5)	4 (5)	19 (32)
61	2 (4)	3 (3)	2 (4)	2 (4)	2 (5)	0 (3)	11 (23)

* - Faulty information due to invalid entries or invalid test completion time

Analysis of the Data

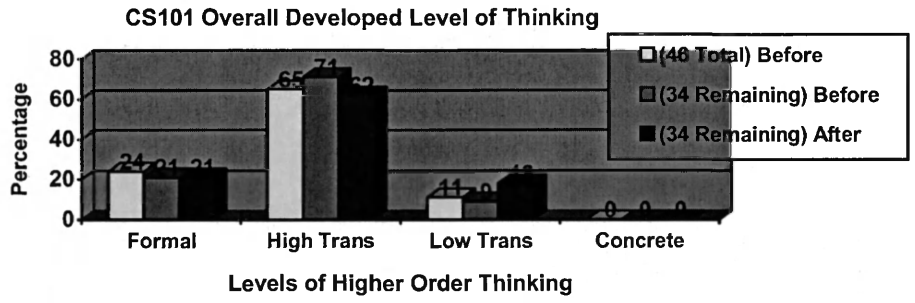
The data collected from the upper level scientific computing and Visual Basic courses are used to assess the students' logical thinking at different levels of computer science courses, predict student outcomes for two types of computer science courses, and compare students' developed formal thinking skills after taking the course. Differences between the students' scientific reasoning skills before and after a course are compared against the experimental programming course. The analysis is used to assess and evaluate new teaching strategies across our introductory level computer science courses that contain students from many different disciplines, inside and outside science. Data reveals that the scientific computing students developed their formal thinking skills prior to entering into CS471, and all the students remained and passed the class that have formal thinking. It isn't surprising that the CS471 students, who are in an upper level college science course, score so high on their logical thinking skills.

Figure 7 shows a graphical representation of students' level of developed thinking before and after taking CS101. The scoring and associated level of thinking is as follows: 78.6% 11-14

(Formal), 50-71.5% 7-10 (High Trans), and 21.4-42.9% 3-6 (Low Trans), 0-14.3% 0-2 (Concrete). This compares to the scoring used in the CS195 case study, which used the Lawson test as a basis. We did not include CS471 because all the students had developed formal thinking prior to taking CS471, and they did not need to be retested. One assumes (and hopes) that students in a senior level science class have either all the skills needed to succeed in scientific courses or developed/acquired the skills as they progress through their scientific discipline.

The figure below shows the skills developed by a typical introduction to computer science course, CS101, at the University of Montana. In an introductory level computer science course, only about 20 –25% of the students have developed their formal thinking skills. These are similar to the results from the experimental programming course, but the percentages for levels below the formal thinking level are very different. The Visual Basic, CS101, course is more representative of the average liberal arts college student wanting to learn about computer science, and these results show that about 11% of those students are in the lowest levels of higher order thinking, which are the concrete and low-level transition stages. As with the experimental course, more students took the test in the beginning of the course than retested at the end of the semester. However, compare the overall level in the beginning of the class to the end of the class, and see the overall level remains roughly the same among the students. A lesser percentage of remaining students are in the formal or low transitional levels, but a greater percentage of the students are in the high transitional level.

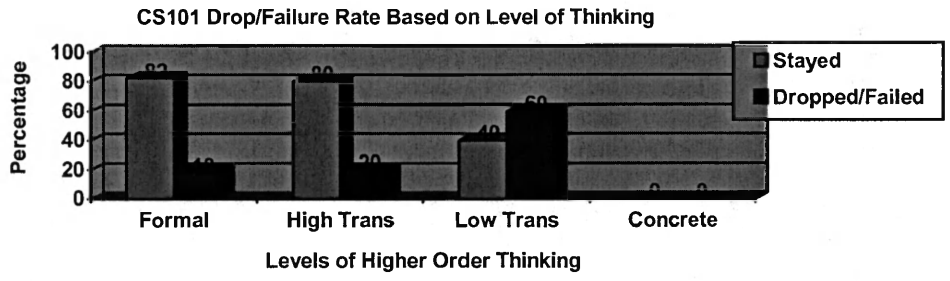
Figure 7



According to CS195 data, there is at least some improvement among the students' overall level of thinking amongst the class. The Visual Basic CS101 course did not show any gain in the remaining students' level of reasoning skills. In fact, it looks as if there is a decline in the overall class's high transitional level and an increased percentage in the low transition level. These results are not as promising as the results for CS195.

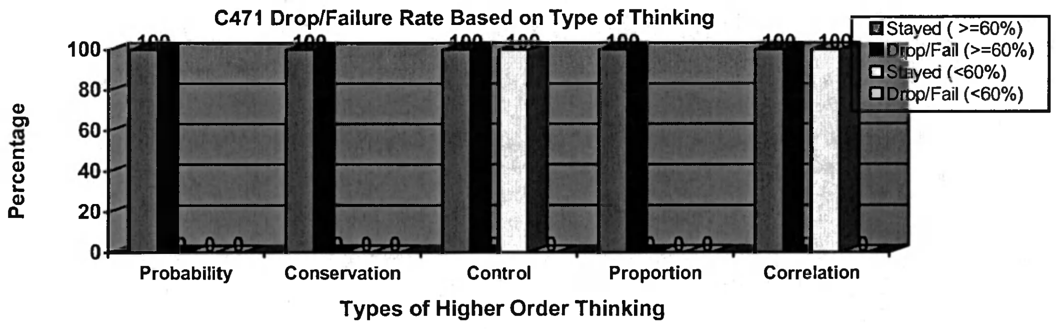
In the figure below, we want to compare the students' level of thinking with their decision or ability to understand a computer science course. In the CS471 class, all the students have formal thinking skills, and all of the students remain in the CS471 class and receive a passing grade. Therefore, we do not analyze CS471 results for dropped/failed status based on thinking type. The next logical reasoning aspect we want to analyze is whether the students' level of thinking or types of thinking skills influences their decision to continue with the class or the grade they receive in the end of the class. According to our results shown below, students that fall into the formal or high transitional levels of thinking are more inclined to stay and pass a course requiring logical thinking versus students that have not even reached the high transitional thinking level.

Figure 8



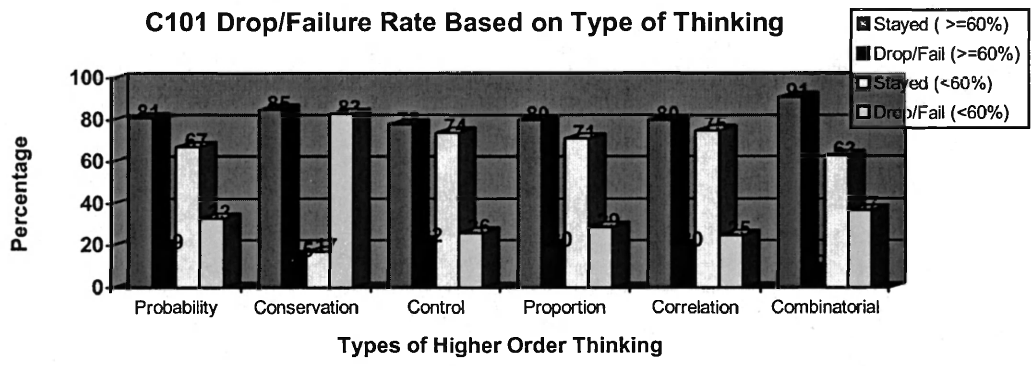
In addition to analyzing the student's overall level of thinking with their performance in the class, we also analyze the individual types of thinking skills and how they compare with the student's performance in the course. Because all of the students in CS471 had formal thinking, most of them have a passing score for each type of thinking skill, except control and correlation reasoning, and those students without these skills still stayed in the class. This differs some from the CS195 results that show all students without control reasoning dropped or failed the course, and the students without correlation skills have a 50/50 chance of passing the course.

Figure 9



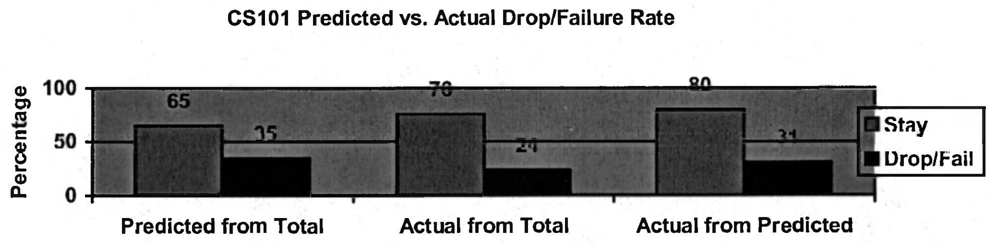
Even though there is a strong relationship between a student's level of thinking and their ability to succeed, Figure 10 does not show such a distinct relationship between the types of thinking skill developed and the student's ability. Also shown by CS101, students have an increased likelihood to drop if they do not have conservation skills, but there is not an indication that students without certain skills will drop/fail. It makes sense that students without conservation skills, which makes up the concrete thinking level, might not be able to handle a course requiring a much higher level of thinking. For more details on the thinking skills, refer to Table 1-1 and Table 1-2.

Figure 10



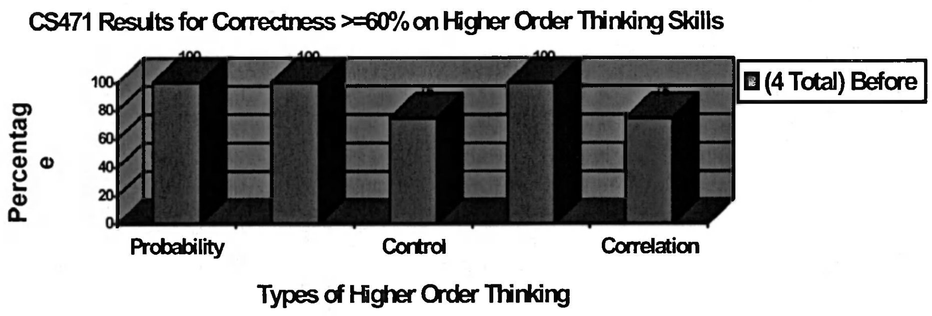
Next, we use the overall level of thinking to predict whether students will drop/fail or stay. The predicted value is based on the student's level of thinking, and if a student has reached the formal level of thinking or fell within the higher end of the high-level transition (9-14 or ≥60%), then he/she is predicted to have the skills needed to stay in the class. We keep this consistent with CS195, but we may have predicted students, whose test scores are above high transitional (7-14 or ≥50%), to stay in the CS101, which is a little lower level class. This is because CS101 was not originally geared to students with interests in science. Of course, all the students in CS471 are predicted to stay in the class and pass because all students have formal thinking. You can see from Figure 11 below, the predicted percentage of students that either stay or drop/fail is very close to the actual percentage of students, who did stay or drop/fail. The last set of results show the actual percentage of students that either stayed or dropped/failed out of the predicted students to stay or drop/fail. The actual percentage of CS101 students that either stayed from those predicted is 80%, but only 30% of those predicted to drop/fail actually did so. This shows a strong correlation between students that have a higher versus a lower level of thinking and their ability to handle a scientific class requiring logical thinking. However, the percentages are not as high as the CS195 results for those predicted to drop/fail out of those who actually did.

Figure 11



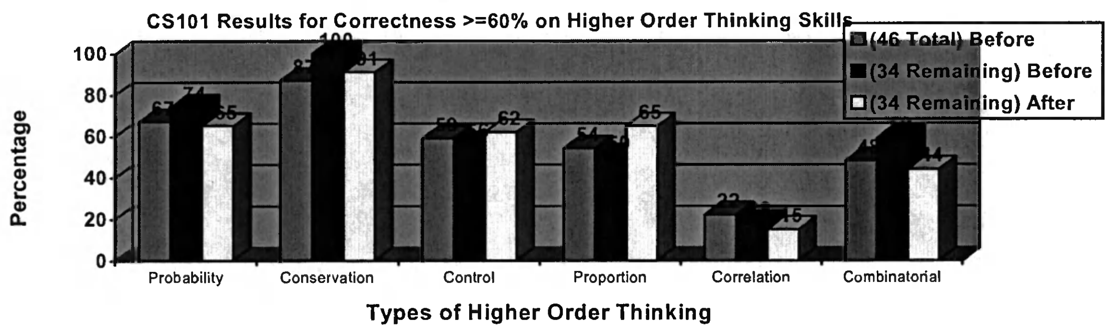
The next set of results show percentages of students that scored greater than or equal to 60% (passing) on individual types of thinking skills. Figure 12 shows results for CS471 students, and control and correlation reasoning skills are the only two skills not at 100% in the class from both the total at the beginning of the course and the remaining students in the class. These students were not retested after CS471, and therefore, we do not have results for these skills after taking CS471.

Figure 12



There is not any improvement in the Visual Basic class's overall level of scientific reasoning, but the individual skills tested in CS101 and their applications to computer science are analyzed. Again, refer to Table 1-1 and Table 1-2 for more details on the higher order thinking skills. Conservation, probability, control, and proportion are the skills that over half the class had developed before CS101. There is an increase in the percentage of remaining students with probability, conservation, and combinatorial skills. This emphasizes the importance of conservation skills and combinatorial skills needed to succeed in a computer science course. The only skills that increase among the remaining students are control and proportion reasoning, and all other skills decline among the Visual Basic students.

Figure 13



Chapter 4

DISCUSSION & FUTURE WORK

This research proves to be a worthwhile experiment in computer science education. The research demonstrates that the higher order thinking skills tests can be used for assessing a student's knowledge and/or evaluating a student and class's improvement over time. The results show a direct correlation between the student's logical reasoning skills and his/her performance in a computer science course. Also, the research shows the overall class and student's learning after taking a class in computer science that should enhance his/her logical thinking ability. Not all computer science classes need to participate in this type of research, because the current research illustrates that the students and classes benefiting the most from computer science education research are the introductory level computer science courses and students. The introductory level area shows the most need for improvement from the overall class and student's scores on the logical thinking tests. Therefore, attention is focused toward the results from the two introductory courses, Experimental Fortran-CS195 and Traditional Visual Basic-CS101, in the discussion and future work.

Discussion

The higher order thinking skills tests can aid our advisors with placing students in the correct computer science class for their ability, and placing a student in the appropriate class for their talents allows the student to grow and gain more confidence to continue. Ideally, a university would have multiple introductory level computer science classes that meet the interests of a variety of students with different learning abilities. This not only reduces the number of students in the introduction to computer science classes, but this solution might capture the interests students have for computers and computer science.

A more realistic approach is to separate the computer science department into theoretical and applied, and therefore, students have a better idea of the type of CS class they are entering and the material covered. There is a reason why such a variety of people take the introduction to computer science course, and the reason is because the students are intrigued by the subject itself and/or they feel the subject can aid them in their discipline of study. Instead of seeing a non-CS student's curiosity for computers being kindled, one tends to see quite the opposite. Very rarely do the students who take a traditional intro-level course, like CS101, take another computer science in the future. Even though computer science is not the discipline of every intro-level computer science student, how many of the students actually reuse their programming skills taught in a course like CS101? How many of those students could benefit from knowing how to combine the computer science skills with their discipline? This is why it is beneficial to know where to place students in computer science courses and have several computer science courses available to meet the variety of interests among students. Also, computer science educators must try to meet the needs of different student's cognitive development using variable learning methods in the classroom. As proposed in this research, this can be accomplished through different ways of promoting active learning in the classroom, such as hands-on lab experiments, class and group discussions, teamwork, etc.

If a plan is made to change the way introductory level computer science courses are taught and organized, then there must be a way to measure the progress of the new learning approach and the student's cognitive development. This research illustrates how scientific reasoning tests are a good measurement of a student's logical thinking level, and therefore, these tests are used as a good indication of a student's improvement after taking a computer science course. Presumably, a student is not going to regress in his/her logical thinking after taking a scientific course, but a student's logical thinking ability may progress because the course helps to develop skills that are not previously existent and/or expresses the already existent skills. In either case, the educator of the science class is doing his/her job if the scores of the student's test results increase. Therefore, the higher order thinking skills test can be used to determine whether the students from a class increase their scientific reasoning skills and increases in students' scores after taking classes with altered pedagogies.

Future Work

There are many directions for this research, but ultimately, this research paves the way for future research using tests for assessing and evaluating computer science education. Currently, a problem exists with capturing and keeping students in introductory level computer science courses, and this research proposes that the future direction of our introduction to computer science courses is toward active and interdisciplinary learning approaches. It may be necessary to split computer science into two separate fields with one being a theoretical approach and the other being an applied program. Computer science is in fact theoretical, and there must remain the study of the science of computers for computer science to continue. A new approach to computer science is needed to accommodate the interdisciplinary and applied nature of CS in today's society. Keeping students current with cutting-edge research assures students are prepared today for their tomorrow.

For a radical movement in computer science education to take place, more research needs to be conducted on measuring the amount that students learn using different teaching strategies in the same course. Even though this research gave results for a variety of computer science classes using different teaching methods, the research lacked varying teaching experiments on the same computer science course to see if there is a difference in the class's improvement employing the different pedagogies into the same class. Conclusions that students benefit more from an active learning approach versus more traditional teaching methods are speculations until more research is completed in this area.

Not only are progressive teaching strategies associated with the future of computer science education, but also change for the introductory level computer science curriculum is envisioned. A movement toward the creation of many introductory level computer science courses and/or separation between theoretical and applied needs to take place to meet the needs and interests of students from a variety of disciplines with a variety of scientific reasoning. Ideally, every discipline should have an introduction to computer science course tailored especially for those students majoring in the specific discipline. Until a critical mass of professors from all the other disciplines become literate in computer programming, the computer science department is going to take the lead in creating the mass of expertise. To emphasize the importance this movement has on the student's learning and career opportunities, more research needs to be conducted on how a student learns differently, what a student does in the future with his/her computer programming knowledge, and whether a student took the traditional introductory level computer science class, like CS101, versus an experimental computer science class tailored to the student's interest, such as CS195.

BIBLIOGRAPHY

Ayen, William E., & Grier, Sam. (1987). A new environment for teaching introductory computer science. *Technical Symposium on Computer Science Education Proceedings of the fourteenth SIGCSE technical symposium on Computer science education*, 258-264.

Biermann, Alan W. (1990). An overview course in academic computer science: a new approach for teaching nonmajors. *Technical Symposium on Computer Science Education Proceedings of the twenty-first SIGCSE technical symposium on Computer science education*, 236-239.

Denning, Peter J., Editor. (1989, December). A debate on teaching computing science. *Communications of the ACM*, 32(12), 1397-1414.

Draves, William A. (1997). *How to Teach Adults, 2nd Ed.* Manhattan, KS: The Learning Resources Network.

Freeman, Peter A. (1997, November). Elements of Effective Computer Science. *Computer*, 22, 76.

Fung, P., O'Shea, T., Goldson, D., Reeves, S., & Bornat, R. (1996, August). Computer Tools to Teach Formal Reasoning. *Computers and Education*, 27(1), 59-69.

Hudak, Mary A, Anderson, David E. (1990, December). Formal Operations and Learning Style Predict Success in Statistics and Computer Science Courses. *Teaching of Psychology*, 17(4), 231-234.

Jehn, Lawrence A., Rine, David C., & Sondak, Norman. (1978). Computer science and engineering education: Current trends, new dimensions and related professional programs. *Technical Symposium on Computer Science Education Proceedings of the ninth SIGCSE technical symposium on Computer science education,* 162-178.

Jones, James S. (1987). Participatory teaching methods in computer science. *Technical Symposium on Computer Science Education Proceedings of the eighteenth SIGCSE technical symposium on Computer science education*, 155-160.

Lawson, A. E. (2000, August). Classroom Test of Scientific Reasoning: Multiple Choice Version, Revised Edition. Tempe: Arizona State University.

Linn, Marcia C. (1995, June). Designing Computer Learning Environments for Engineering and Computer Science: The Scaffolded Knowledge Integration Framework. *Journal of Science Education and Technology*, 4(2), 103-126.

Magel, Kenneth. (1989, February). Discontinuities in Computer Education. *Computer*, 30, 48-48.

Monteyne, K., Cracolice, M. (2002). The Development and Validation of a Web-based Assessment of Higher-Order Thinking Skills. Manuscript in preparation, The University of Montana.

National Commission on Excellence in Education. (1983). *A Nation at Risk.* Washington, DC: U.S. Government Printing Office.

National Science Board of Education. (2002). *Science and Engineering 2002 Indicators.* Retrieved October 20, 2002, from http://www.nsf.edu/sbe/srs/seind02

Negal, Greta. (1994). *The Tao of Teaching.* New York: Donald I. Fine, Inc.

Pigford, D. V. (2001, December). Designing and implementing active learning in the computer science curriculum: an interactive tutorial. *The Journal of Computing in Small Colleges,* 17(2), 199-204.

Priebe, Rodger. (1997, March). The Effects of Cooperative Learning in a Second-Semester University Computer Science Course. ERIC ED406189.

Richmond, P. G. (1970). *An Introduction to Piaget.* New York: Basic Books, Inc.

Roadrangka, V., Yeany, R. H., & Padilla, M. J. (1983). The Construction and Validation of Group Assessment of Logical Thinking (Galt). Paper presented at the Annual meeting of the National Association for Research in Science Teaching, Dallas, TX.

Rodger, Susan H. (1995). An interactive lecture approach to teaching computer science. *Technical Symposium on Computer Science Education Proceedings of the twenty-sixth SIGCSE technical symposium on Computer science education,* 278-282.

Schwill, Andreas. (1997, September). Fundamental Ideas: Rethinking Computer Science Education. *Learning and Leading with Technology,* 25(1), 28-31.

Tobin, K. G., & Capie, W. (1981). The Development and Validation of a Group Test of Logical Thinking. *Educational and Psychological Measurement,* 41(2), 413-423.

University of Montana. (2002). *Higher Order Thinking Skills (HOTS) Test.* Retrieved May 31, 2003, from http://hots.chem.umt.edu

Wees, W. R. (1971). *Nobody Can Teach Anyone Anything.* Garden City, NY: Doubleday and Company, Inc.

Wileman, Stanley, Konvalina, John, & Stephens, Larry J. (1981, March-April). Factors Influencing Success in Beginning Computer Science Courses. *Journal of Educational Research,* 74(4), 223-226.

Wilson, Kenneth G., & Daviss, Bennett. (1994). *Redesigning Education.* New York: Henry Holt and Company.

Appendix A: CS195 Syllabus & Coursework

Computer Science 195 - Spring 2002

Introduction to Computational Science

Prerequisites

- Co-requisite - MATH 100

This course is aimed at students interested in science outside of computer science, with focus on the development of scientific codes using the FORTRAN programming language.

Objectives

This course is designed to introduce college students to the interdisciplinary uses of math, programming, and computer visualization used in the sciences. The intent of the class is to show students the uses of computers and mathematical skills needed outside the computer science environment. The focus of this course is on the computation and computer programming used to advance research in the scientific community. The course will introduce elementary programming techniques using the Fortran programming language in a Unix environment and mathematical/scientific computer programs currently used to enhance scientific research.

The course will involve interaction with scientists around the nation through the use of the Access Grid Node resources now present at The University of Montana (http://mroccs.cs.umt.edu/AGN/).

Instructor Information

Jennifer Parham
Social Sciences 423 B
Email: cs195@sheba.cs.umt.edu

Class Website: mroccs.cs.umt.edu/cs195

Office Hours: 2-5, (M, W)
Drop in/Appointment (T, TH, F)

Class Meeting Times/Place

- 1100 - 1200 MWF
- Social Science 362

Attendance Policy

Class attendance is not a factor in determining grades. When a class is missed, it is the STUDENT'S responsibility to obtain any notes, assignments, etc. from classmates.

Required Text

Introduction to Fortran 90 For Engineers and Scientists, Larry R. Nyhoff, Sanford C. Leestma, Prentice Hall, ISBN 0-13-505215-7

Sams Teach Yourself UNIX in 24 Hours, Second Edition, Dave Taylor and James C. Armstrong, Jr., Sams Publishing, ISBN 0-672-31480-0

Grade Evaluation

6-8 assignments - 50%

- Programming assignments will *NOT* be accepted after the stated deadlines.
- Program assignments which do not compile will not be graded.
- In general, no extensions of program deadlines - plan ahead and anticipate system outages, etc.

weekly quizzes - 10%

- A weekly quiz will be given almost every Monday to ensure that all students are learning with little difficulty and the teaching is conveying the message clearly.
- The quizzes will be very short, and they will cover material from the previous week.
- The lowest quiz grade will be dropped, and no make up quizzes will be given.

weekly labs - 10%

- Labs are designed to be finished during Friday classtime (30-45min), but may be done or finished at home.
- Labs are graded on effort not right or wrong. If you make an effort to complete the lab, you will receive an A.
- These labs are used to enhance your lectures using hands-on learning.

Two exams - 30% (15% each exam)

- Must notify instructor *BEFORE* the exam to schedule a makeup.
- Midterm Exam - Monday, March 11, 2002 (subject to change)
- Final Exam - During Finals Week (possibly a project)

Grading Scale

Grade	**Average**
A	**90 or greater**
B	**80-89**
C	**70-79**
D	**60-69**
F	**less than 60**

Tentative Course Topics/Schedule

Meeting Date	**Class Description**
	WEEK 1 - Introduction
January 28	Introduction/Higher Order Thinking Skills Test
January 30	Programming/FORTRAN Introduction
Feburary 01	Lab #1 - Login and Unix Worksheet
	WEEK 2 – Area Code and More Unix - Assignment #1 DUE
Feburary 04	Area, Design Flow Chart and Pseudocode for Area Code
Feburary 06	Continue Developing Area Code in FORTRAN
Feburary 08	Lab #2 - Compile/Link/Run Area Source Code
	WEEK 3 – Machine Precision
Feburary 11	Scientific #s and Machine Precision
Feburary 13	Underflow and Overflow / Write Machine Precision Code in FORTRAN
Feburary 15	Lab #3 - Run Experiments with Machine Precision Code
	WEEK 4 – Error Assessment - Assignment #2 DUE
Feburary 18	**** PRESIDENT'S DAY **** NO CLASS ****
Feburary 20	Actual and Relative Error
Feburary 22	Iteration Until Convergence in FORTRAN
	WEEK 5 – Integration
Feburary 25	Integration
Feburary 27	Conditional Operators / Built in FunctionsIteration Until Convergence in FORTRAN
March 01	Lab #4 - Debug FORTRAN Code Using a Graphical Debugger
	WEEK 6 – Derivatives - Assignment #3 DUE
March 04	Derivatives (Partial and Full)
March 06	Vectors and Arrays
March 08	Lab #5 - Introduction to Maple / Review for Midterm Exam
	WEEK 7 Midterm Testing
March 11	MIDTERM EXAM
March 13	Review Test Answers
March 15	NO CLASS, HAVE A NICE BREAK:)
	WEEK 8 ** SPRING BREAK ********
March 18	NO CLASS
March 20	NO CLASS

March 22	NO CLASS
	WEEK 9 Vectors and Visualization
March 25	Review / Questions on 1st Half of Semester
March 27	Dervatives and Vectors in FORTRAN
March 29	Lab #6 - Introduction to GrADS Visualization
	WEEK 10 Databases – Assignment #4 DUE
April 01	Introduction to Databases
April 03	Databases in Fortran
April 05	Lab #7 - More GrADS / Visualization and Databases
	WEEK 11 – Systems of Equations
April 08	Introduction to Systems of Equations
April 10	Matrices
April 12	Lab #8 - Linear Algebra with Maple
	WEEK 12 – Heat Diffusion - Assignment #5 DUE
April 15	Solving Systems of Equations (Explicit vs. Implicit)
April 17	Explicit Method in FORTRAN
April 19	Lab #9 - GrADS and Maple for Visualization
	WEEK 13 - Work on N-body Problem
April 22	Develop N-body Code
April 24	Continue Developing N-body Code
April 26	Lab #10 - Linking / Makefiles / Timing Programs
	WEEK 14 - Supercomputing - Assignment #6 DUE
April 29	Introduce Supercomputing / Parallelism
May 01	More Supercomputing
May 03	Lab #11 - Biology Workbench / Tera Grid
	WEEK 15 - Wrap Up - Assignment #7 DUE
May 06	Finish Up
May 08	Retest Higher Order Thinking Skills
May 10	Review for Final

CS195 - Introduction to Computational Science

LECTURE NOTES & LAB

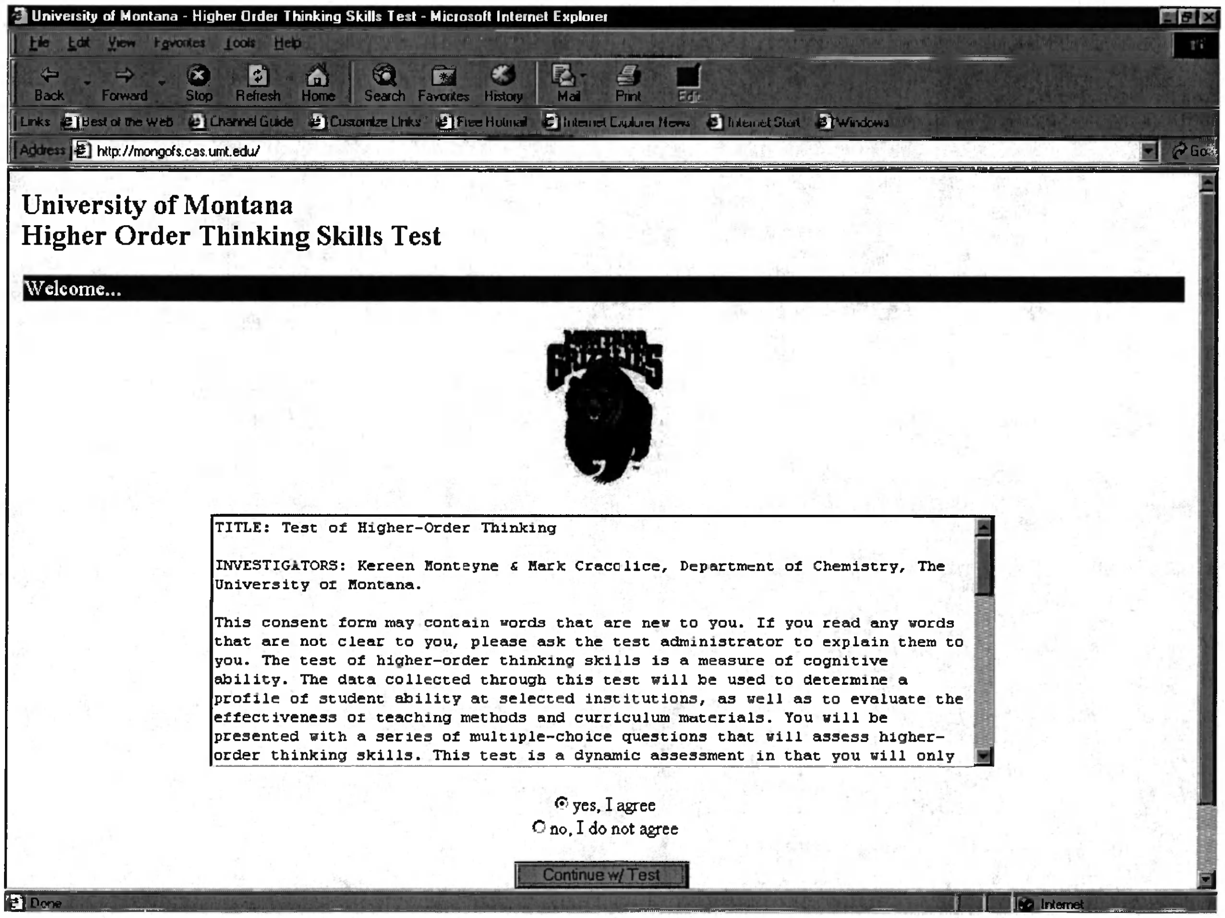
1/28 - Syllabus / Introduction to Computational Science
1/30 - Introduction to Computer Programming
2/01 - Introduction to Unix
  Lab #1 - Login / Unix Commands
2/04 - Higher Order Thinking Test
2/06 - Designing a Program / Psuedocode and Flowchart
2/08 - Go Over Lab #1 / More Unix
  Lab #2 - Compile / Link Area Program
2/11 - FORTRAN90 Program Syntax
2/13 - More FORTRAN90 Program Syntax
2/15 - Lab #3 - UNIX Environment / Paths
2/18 - President's Day - NO CLASS
2/20 - Scientific Numbers & Machine Precision
2/22 - Lab #4 - Experiment w/ Machine Precision Code
2/25 - Error Assessment
2/27 - Develop Convergence Code / FORTRAN90 Loops
3/01 - Lab #5 - Program Errors and Debugging Code
3/04 - Integrating Functions
3/06 - Revisit Integration & Approx. Methods
3/08 - FORTRAN90 Functions and Subroutines
3/11 - More Functions and Subroutines
3/13 - Review / Questions
3/15 - Spring Break - NO CLASS
3/25 - Review / Quiz
3/27 - Derivatives / Arrays
3/29 - Lab #6 - Octave / Math program
4/01 - Finish 1-D Arrays
4/03 - Input & Output/Databases
4/05 - Lab #7 - GrADS Gridded Analysis and Display System
4/08 - Science Fair / Quiz
4/10 - Finish Databases/2-D Matrices
4/12 - Lab #8 - More GrADS Visualization & Databases
4/15 - 1-D Heat Diffusion
4/17 - Explicit Heat Diffusion
4/19 - Arctic Region Supercomputing Center (ARSC)
4/22 - Finish 1-D Heat Diffusion/Implicit Method
4/24 - National Computational Science Alliance (NCSA)
4/26 - Unfolding Universe - Donna Cox (UIC)
4/29 - Lab #9 - Linking Programs / Makefiles
5/01 - High Performance Computing / Parallel Processing
  Using Summit Beowulf
5/03 - Lab #10 - Running Parallel Code
5/06 - Final Review
5/08 - Higher Order Thinking Skills Re-Test
5/10 - Review / Questions

FINAL
DUE 5/17 - FINAL - Pick a project
MIDTERM
DUE 3/25 - Chap. 6/7 - MIDTERM - 1-D Integration

ASSIGNMENTS
DUE 2/08 - Read Chap. 1 - Fortran90 for Scientists
Assignment #1 - Computational Science
DUE 2/15 - Read Chap. 2 - Fortran90 for Scientists
Assignment #2 - Re-write Area Program as Volume Program
DUE 3/04 - Read Chap. 3 - Fortran90 for Scientists
Assignment #3 - Re-Write Volume Program to accept a box or sphere
DUE 3/11 - Read Chap. 4 - Fortran90 for Scientists
DUE 3/25 - Read Chap. 6 & 7 - Fortran90 for Scientists
DUE 4/03 - Read Chap. 8.1 - Fortran90 for Scientists
Assignment #4 - Re-Write Integration Program to use Arrays
DUE 4/08 - Read Chap. 8.2, 8.3, 5 - Fortran90 for Scientists
DUE 4/19 - Read Grads Tutorial & Fortran90 Arrays and Input/Output
Assignment #5 - Construct a GrADS data set and visulize data
DUE 5/08 - Assignment #6 - Calculate 1-D Heat Diffusion Using Implicit Method

Appendix B: UM's HOTS Test User Guide

Higher Order Thinking Skills Test


University of Montana - Higher Order Thinking Skills Test - Microsoft Internet Explorer
http://mongofs.cas.umt.edu/

University of Montana
Higher Order Thinking Skills Test

Welcome...

TITLE: Test of Higher-Order Thinking

INVESTIGATORS: Kereen Monteyne & Mark Cracolice, Department of Chemistry, The University of Montana.

This consent form may contain words that are new to you. If you read any words that are not clear to you, please ask the test administrator to explain them to you. The test of higher-order thinking skills is a measure of cognitive ability. The data collected through this test will be used to determine a profile of student ability at selected institutions, as well as to evaluate the effectiveness of teaching methods and curriculum materials. You will be presented with a series of multiple-choice questions that will assess higher-order thinking skills. This test is a dynamic assessment in that you will only

[✓ yes, I agree]
no, I do not agree

Continue w/ Test

TABLE OF Contents

What is the HOTS Test?

HOTS stands for ***Higher Order Thinking Skills.*** The HOTS test is an online testing tool designed for the University of Montana school system. This test will be administered to all UM incoming freshman students as well as selected classes.

Originally, this test was designed as a clinical test, which meant the test was given to students one by one with a specialist present. This method of testing is very time consuming, especially for a large set of test subjects. Therefore, an online version of this Higher Order Thinking Skills Test will allow for easy testing of all new university students during orientation and all exiting students before graduation.

Major Features

Major features of the online HOTS test include:

- Personal Information
- Missing Information
- Test Instructions
- Taking the Test
- Student Results
- Database Maintenance
- Interface for adding/deleting/modifying test questions

How does it Work?

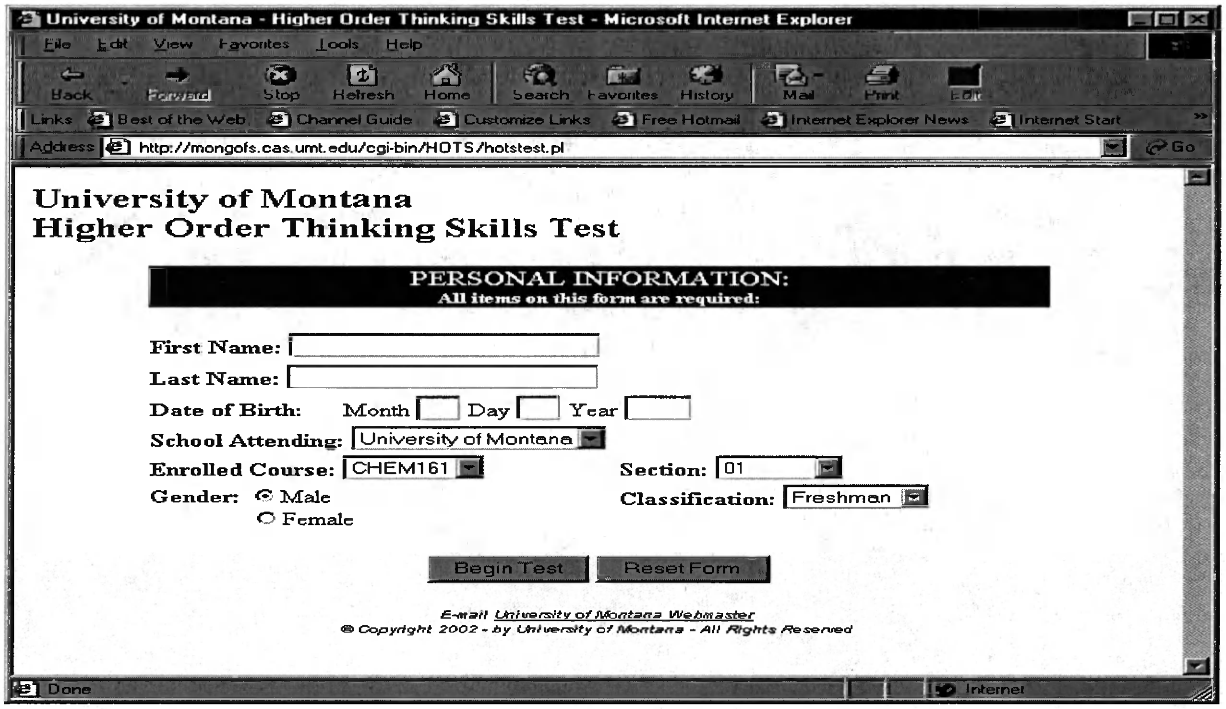
Students can access the online HOTS Test through a standard web browser by entering the URL, web address. The students will be asked to enter their personal information, such as social security number, name, etc. Once they have entered all required information into all the fields on the form, they can proceed with taking the test.

The test will consist easy, medium, and hard questions, which will be picked beginning with a medium question from the databases, which each contains questions of a different thinking skill type. Each question is either graded as an individual question or as two separate questions, depending on the type of the question. The user will select the answers he/she feels are correct, and then the user will click a button to proceed with the next question until all thinking skill types have been tested.

HOTS will then calculate the time it took the student to complete the test and the skill level for each thinking type. This information, as well as a log of the correctness of each question answer, is recorded in a database for later querying.
Administrators will be able to add and delete questions from the test question database through an online form. Also, the administrators will be responsible for putting the pictures that go with test questions into their appropriate jpeg format.

Personal Information

Before taking the test, the student is required to enter information about themselves and their status at the University.


University of Montana - Higher Order Thinking Skills Test - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Links Best of the Web Channel Guide Customize Links Free Hotmail Internet Explorer News Internet Start
Address http://mongofs.cas.umt.edu/cgi-bin/HOTS/hotstest.pl
Go
Done
Internet

University of Montana
Higher Order Thinking Skills Test

PERSONAL INFORMATION:
All items on this form are required:

First Name:
Last Name:
Date of Birth: Month Day Year
School Attending: University of Montana
Enrolled Course: CHEM161 **Section:** 01
Gender: [✓ Male] Female **Classification:** Freshman

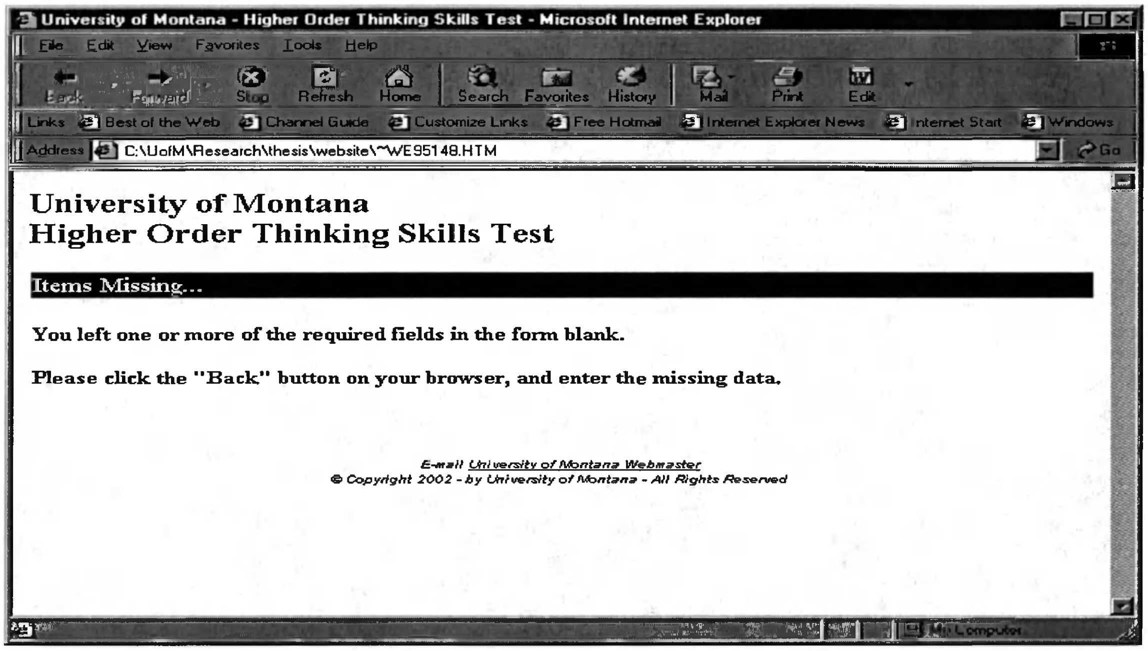
Begin Test | Reset Form

E-mail University of Montana Webmaster
© Copyright 2002 - by University of Montana - All Rights Reserved

The required information is as follows:

- First Name
- Last Name
- Date of Birth
- School Attending
- Enrolled Course and Section
- Gender
- Classification at the University

The personal information form must be completed with the appropriate information before continuing with the test. If any of the personal information is left blank, then the user is presented with the following missing information screen.


University of Montana - Higher Order Thinking Skills Test - Microsoft Internet Explorer
Address C:\UofM\Research\thesis\website\~WE95148.HTM

University of Montana
Higher Order Thinking Skills Test

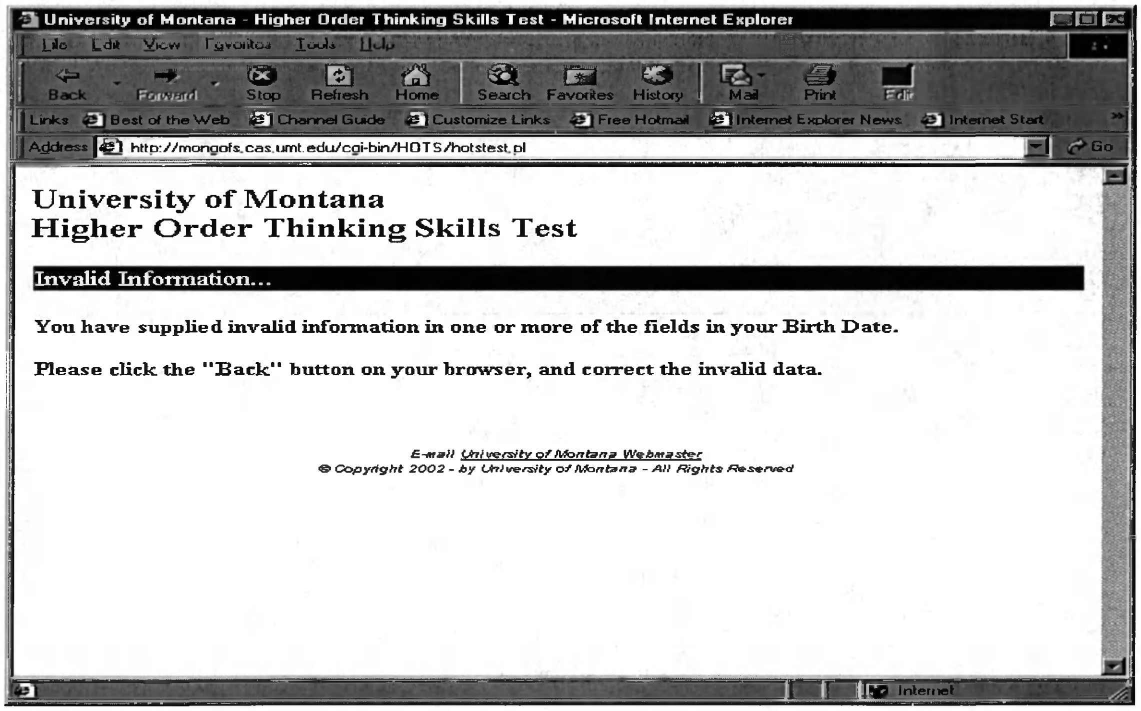
Items Missing...

You left one or more of the required fields in the form blank.

Please click the "Back" button on your browser, and enter the missing data.

E-mail University of Montana Webmaster
© Copyright 2002 - by University of Montana - All Rights Reserved

The user must enter a valid birth date, and if an invalid birth date is entered, then an invalid information page is displayed. In which case, the user must press the back button and enter a valid birth date to continue.


University of Montana - Higher Order Thinking Skills Test - Microsoft Internet Explorer
Address http://mongofs.cas.umt.edu/cgi-bin/HOTS/hotstest.pl

University of Montana
Higher Order Thinking Skills Test

Invalid Information...

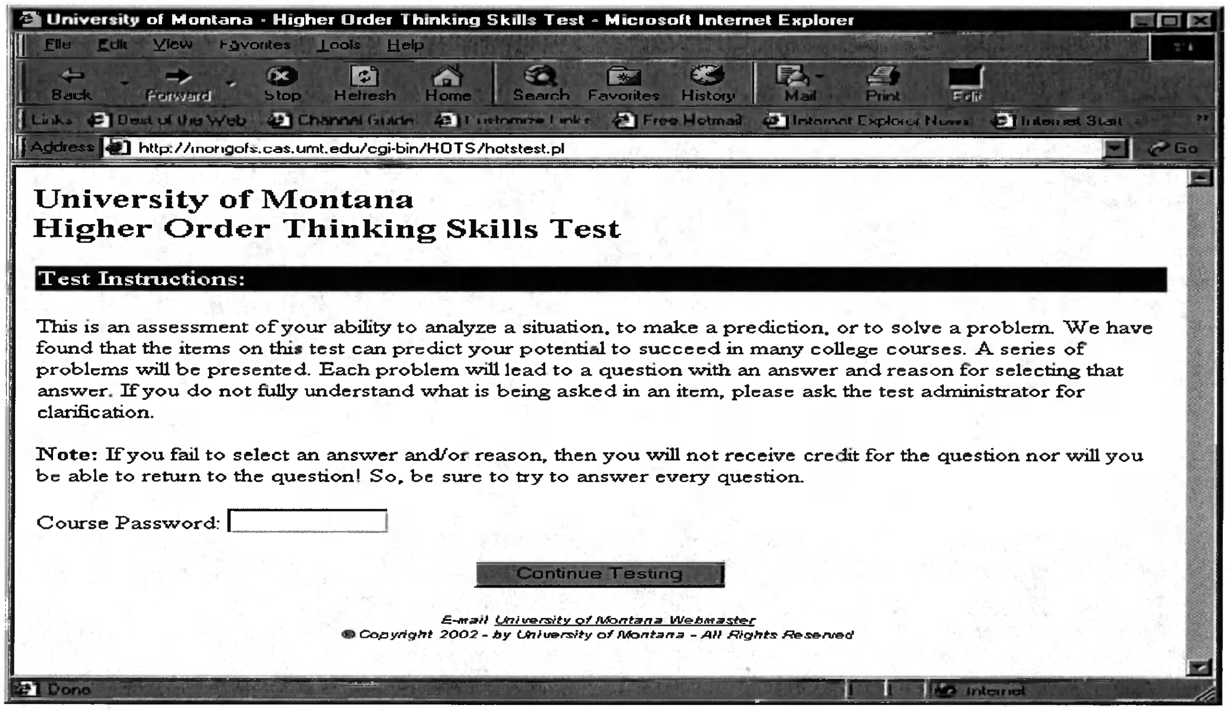
You have supplied invalid information in one or more of the fields in your Birth Date.

Please click the "Back" button on your browser, and correct the invalid data.

E-mail University of Montana Webmaster
© Copyright 2002 - by University of Montana - All Rights Reserved

Testing Instructions

The Higher Order Thinking Skills Test instructions are presented after the student has entered his/her personal information. These instructions are used to inform the student of what is included in the HOTS Test and how to approach each question. The student is encouraged to answer every question, and the questions left blank are counted as incorrect. The student is asked to supply a password for his/her class. This ensures that the student cannot go to the site at any time and retake the test.


University of Montana - Higher Order Thinking Skills Test - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Address http://mongofs.cas.umt.edu/cgi-bin/HOTS/hotstest.pl
Go
Continue Testing
Done
Internet

University of Montana
Higher Order Thinking Skills Test

Test Instructions:

This is an assessment of your ability to analyze a situation, to make a prediction, or to solve a problem. We have found that the items on this test can predict your potential to succeed in many college courses. A series of problems will be presented. Each problem will lead to a question with an answer and reason for selecting that answer. If you do not fully understand what is being asked in an item, please ask the test administrator for clarification.

Note: If you fail to select an answer and/or reason, then you will not receive credit for the question nor will you be able to return to the question! So, be sure to try to answer every question.

Course Password:

E-mail University of Montana Webmaster
© Copyright 2002 - by University of Montana - All Rights Reserved

Note: Once a question has been either answered or left blank, the student cannot go back to the previous questions and change answers.

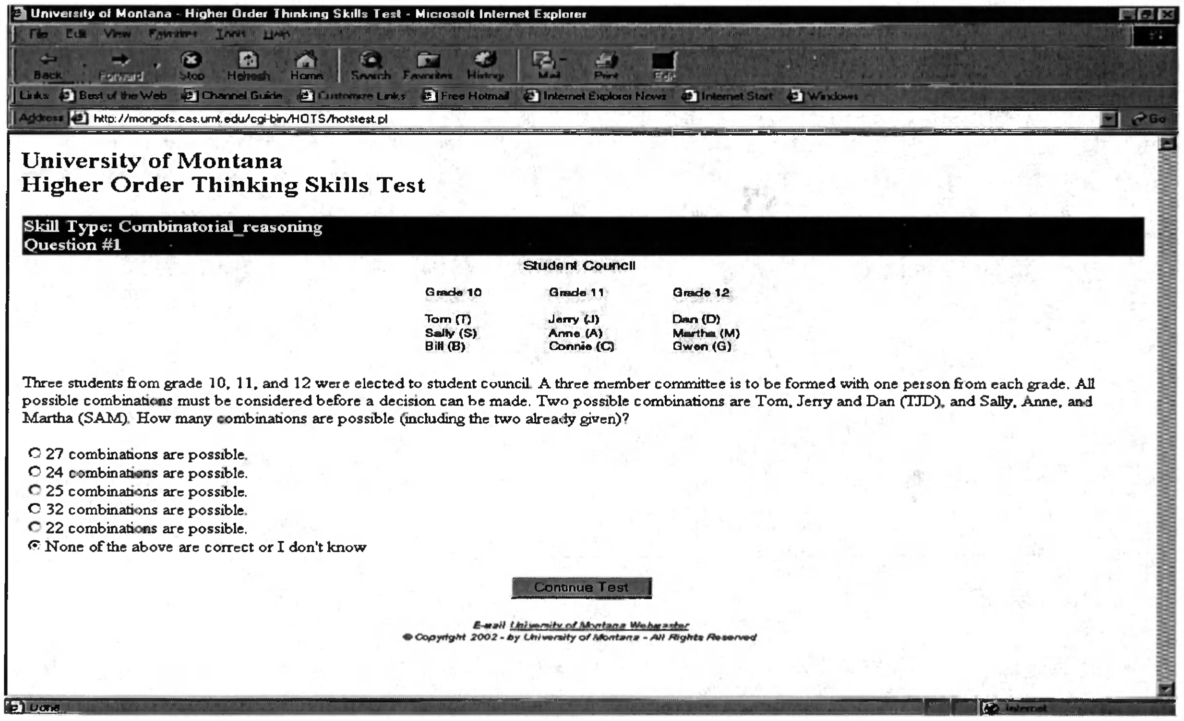
Taking the Test

Once the test has begun, after the instructions page is presented, the initial time is recorded. Then after the student has finished the test, the final time is also recorded and the total test time is calculated.

The student is presented a medium level question from a random thinking skill type database. The student can answer the question or leave the question as the selected default option, which is "I don't know", but the default answer is considered incorrect. If the student does not answer the medium level question correctly, then he/she will be presented with a question from an easy level. However, if the student answers the medium level question correctly, then he/she will be presented with a hard level question. After the student answers two questions correctly from a specific level of thinking without failing two of that type, then he/she advances to the next type of thinking skill.

Most questions will consist of a question and a reason, where both parts of the question must be answered correctly to receive full credit. Other questions from the hypothetico-deductive reasoning types consist of two parts, but each part (question and reason) count as separate questions. Therefore, these two-part questions count as two correct answers in the specific level. Also, there are questions from both, hypothetico-deductive and combinatorial reasoning, that consist of only one part to the question, and therefore, just that part must be answered to receive credit.

Each question in each database corresponding to a different thinking type has a difficulty level and question id associated with it. The very first question presented to the student is a medium level question, but it may come from any of the skill types. After the initial question, the student is presented with a series of unrepeated questions based on the difficulty level of the students' performance. There are three difficulty levels, Easy, Medium, and Hard, and the student is advanced to a harder level if he/she answers two questions of the same level correct.



http://mongofs.cas.umt.edu/cgi-bin/HOTS/hotstest.pl

University of Montana
Higher Order Thinking Skills Test

Skill Type: Combinatorial_reasoning
Question #1

Student Council

Grade 10	Grade 11	Grade 12
Tom (T)	Jerry (J)	Dan (D)
Sally (S)	Anne (A)	Martha (M)
Bill (B)	Connie (C)	Gwen (G)

Three students from grade 10, 11, and 12 were elected to student council. A three member committee is to be formed with one person from each grade. All possible combinations must be considered before a decision can be made. Two possible combinations are Tom, Jerry and Dan (TJD), and Sally, Anne, and Martha (SAM). How many combinations are possible (including the two already given)?

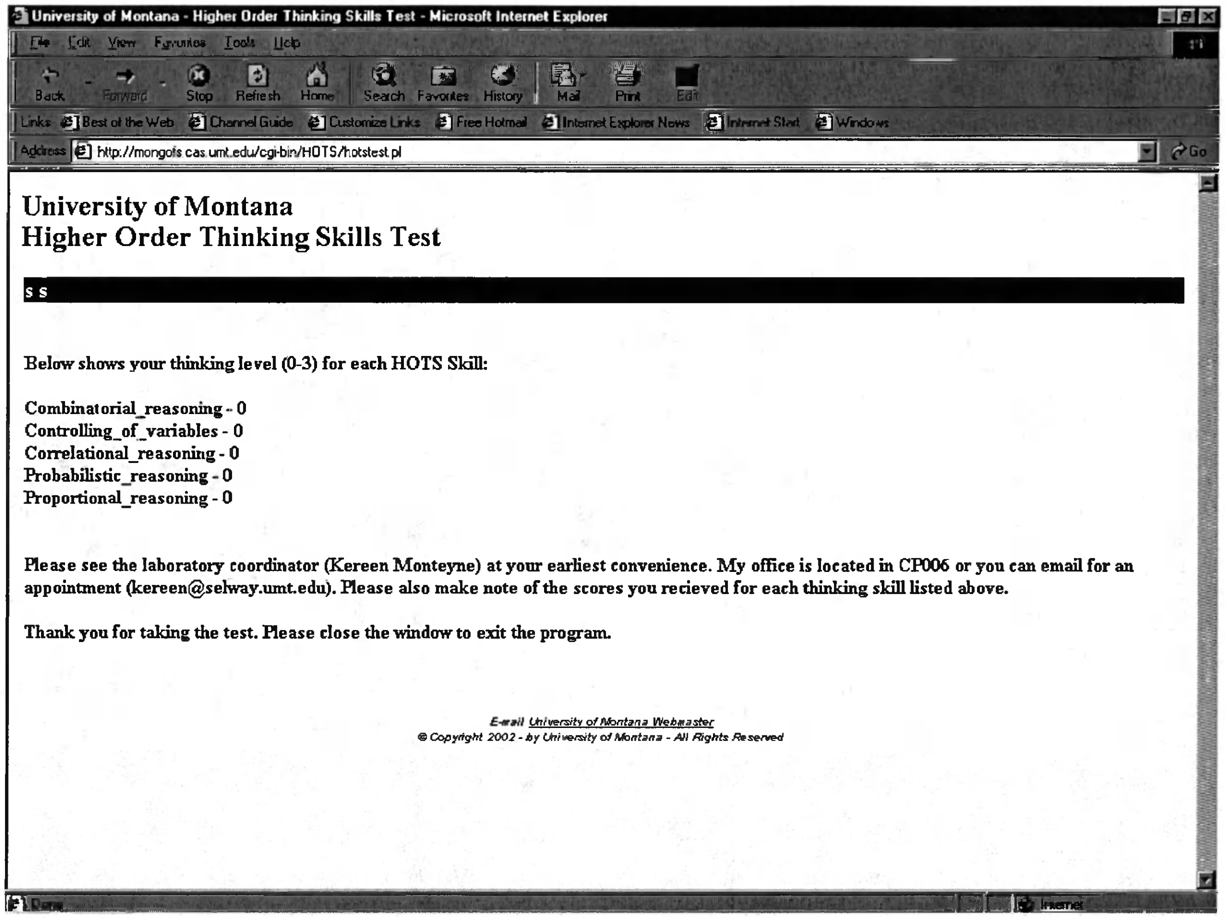
- 27 combinations are possible.
- 24 combinations are possible.
- 25 combinations are possible.
- 32 combinations are possible.
- 22 combinations are possible.
- [✓ None of the above are correct or I don't know]

E-mail University of Montana Webmaster
© Copyright 2002 - by University of Montana - All Rights Reserved

Note: The student cannot go back to the previous questions and change answers.

Student Results

After all questions in the test have been answered, a results page is displayed for the student. This page is used to thank the student for taking the test and let the student know he / she has finished the test. This page also informs the student of their level in each skill type on the test. When the student results are displayed, all the student information is saved to the database as well.


University of Montana - Higher Order Thinking Skills Test - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Links Best of the Web Channel Guide Customize Links Free Hotmail Internet Explorer News Internet Start Windows
Address http://mongofs.cas.umt.edu/cgi-bin/HOTS/hotstest.pl
Go

University of Montana
Higher Order Thinking Skills Test

s s

Below shows your thinking level (0-3) for each HOTS Skill:

Combinatorial_reasoning - 0
Controlling_of_variables - 0
Correlational_reasoning - 0
Probabilistic_reasoning - 0
Proportional_reasoning - 0

Please see the laboratory coordinator (Kereen Monteyne) at your earliest convenience. My office is located in CP006 or you can email for an appointment (kereen@selway.umt.edu). Please also make note of the scores you recieved for each thinking skill listed above.

Thank you for taking the test. Please close the window to exit the program.

E-mail University of Montana Webmaster
© Copyright 2002 - by University of Montana - All Rights Reserved

Database Maintenance

The database contains all of the student's personal information, date, test question information and test question correctness, total percentage of correct answers on the test, and time to complete the test. The database is text, field delimited by pipes (|), and the database is able to be imported to any database manager, such as Access, SQL, etc., by using the pipe (|) as the delimiter. The databases correspond to student's scores from specific courses.

Once the data has been imported to the database manager, the manager can be used to query the database for other useful information, such as the average time it took to take the test for a specific year, percentage of students above a specific level for a specific skill, etc.
Last | First | Course | Birthdate | Gender | Class | Test Date | Skill Type | Student_level | … | ques1_level | ques1_type | ques_id | student_answer| student_reason | correct_answer | ... | Test Time

Doe | John | CS101 | 09/05/1970 | M | Soph | 07/15/2002 | Conservation | 2 | … | Medium | Conservation | CONSM01 | 2 | 1 | 1 | 1 | ... | 1.30

There will also be many databases corresponding to the different types of skills. Each database will contain questions and answers to choose from, and the questions in the database will contain the corresponding reasons for choosing the answer for that question. Having many databases eliminates the need to search one large database for the different types of questions for the test. However, the level of difficulty of the question will be added to the database, so inevitably there will be sorting and searching the question database. Below is an example of the question database:

Question 1 Level | Question 1 Type | Question ID | Question 1 | #answers | answer1 | answer2 | answer3 | ... | correct answer | reason | #reasons | reason1 | reason2 | ... | correct reason | picturename.jpg

Easy | Conservation of mass | CONSE01 | Suppose you are given two clay balls of equal size and shape. The two clay balls also weigh the same. One ball is flattened into a pancake-shaped piece. Which of these statements is correct? | 3 | The pancake-shaped piece weighs more than the ball | The two pieces still weigh the same | The ball weighs more than the pancake-shaped piece | 2 | because | 5 | the flattened piece covers a larger area | the ball pushes down more on one spot | when something is flattened it loses weight | clay has not been added or taken away | when something is flattened it gains weight | 3 | 730582002.jpg

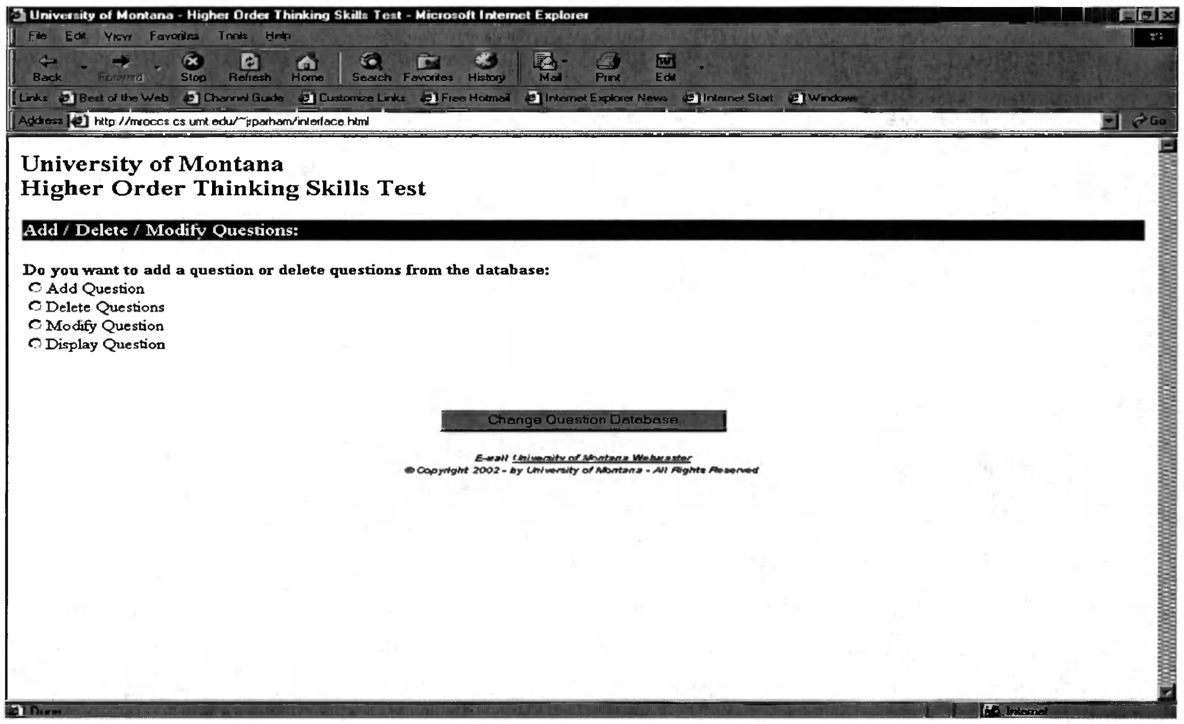
The database is sorted from easy to hard, with a beginning line in the database indicating the number of questions in each level. These numbers are separated by spaces, for example:

#Easy #Medium #Hard
12 10 5

The pictures are named according to the time and date when they were uploaded, i.e. timedate.jpg. For example, if a picture was uploaded at 7:30am, May 15, 2002, then the picture name in the database is 7305152002.jpg.

Interface to Add/Delete/Modify/View Questions

The web interface for adding, deleting, modifying, and viewing questions from the database is designed to ease the job of updating the questions for the administrators of the testing web site. The administrator is asked whether he/she wants to add, delete, modify, or view a question in the database. Then depending on the action he/she wants to take, a new interface will appear.


University of Montana
Higher Order Thinking Skills Test

Add / Delete / Modify Questions:

Do you want to add a question or delete questions from the database:

- Add Question
- Delete Questions
- Modify Question
- Display Question

Change Question Database

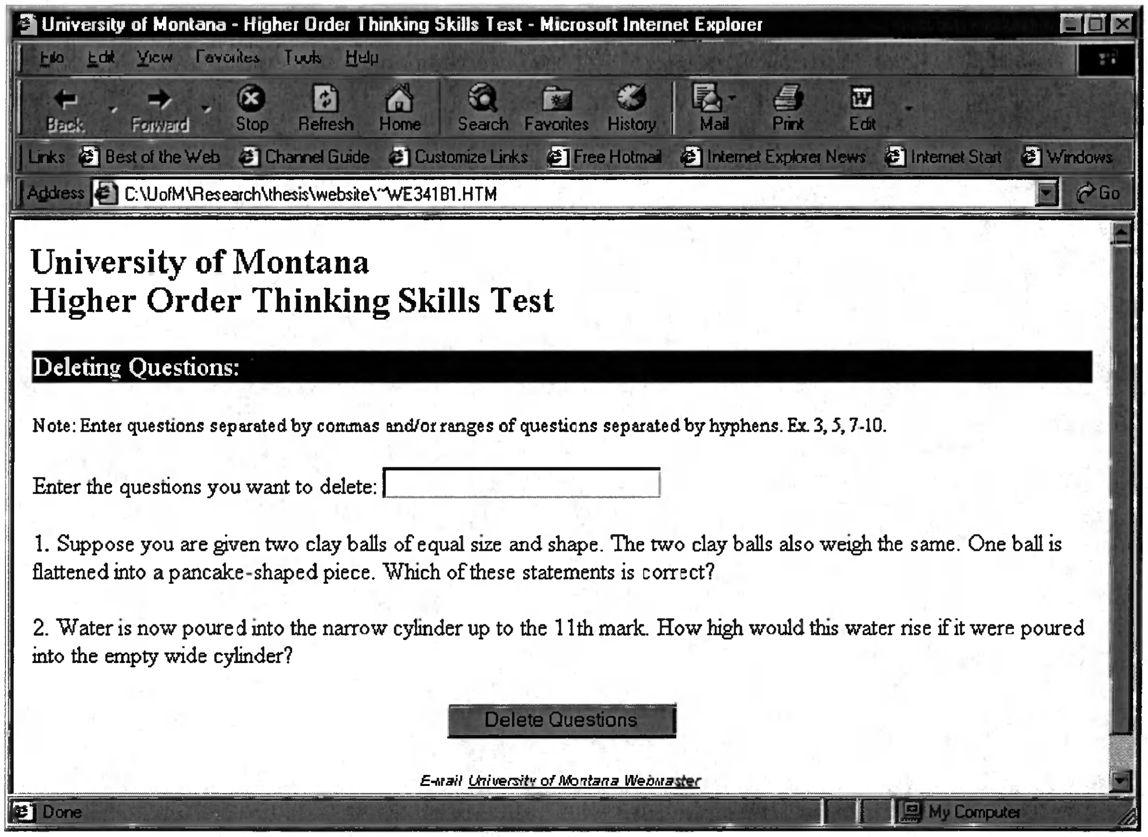
E-mail University of Montana Webmaster
© Copyright 2002 - by University of Montana - All Rights Reserved

Deleting a Question:

If the administrator wants to delete a question, he/she will be presented with a list of the questions from both databases that he/she is allowed to delete. The user is asked to enter the questions he/she would like to delete, and formal instructions are supplied to help the user choose from a range of questions or multiple, individual questions to delete. The user enters a range of questions using the hyphen, -, and separate questions by using a comma.

The user must enter the range of questions in increasing order, but the individual questions can come in any order. However, if the user makes a mistake and enters a number not within the range of numbers in the database or a range in decreasing order, then an invalid range of questions page is presented. Also, if the user enters a range to delete that spans across multiple databases, then an error page is presented.

Once the administrator has selected a valid question for deletion and pressed the "Delete Questions" button, the question is completely removed from the database. The associated picture is also removed from the server, and the number of questions in the same difficulty level is decremented.


University of Montana - Higher Order Thinking Skills Test - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Links Best of the Web Channel Guide Customize Links Free Hotmail Internet Explorer News Internet Start Windows
Address C:\UofM\Research\thesis\website\~WE341B1.HTM
Go

University of Montana
Higher Order Thinking Skills Test

Deleting Questions:

Note: Enter questions separated by commas and/or ranges of questions separated by hyphens. Ex. 3, 5, 7-10.

Enter the questions you want to delete:

1. Suppose you are given two clay balls of equal size and shape. The two clay balls also weigh the same. One ball is flattened into a pancake-shaped piece. Which of these statements is correct?

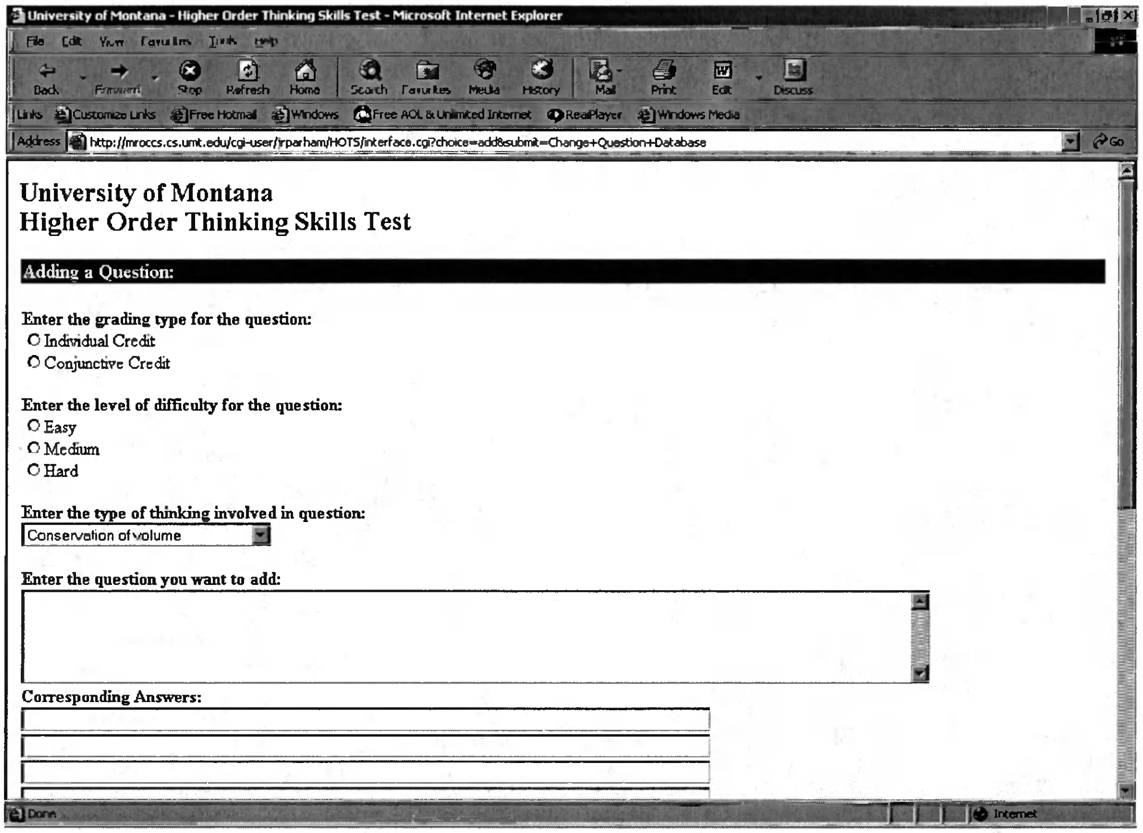
2. Water is now poured into the narrow cylinder up to the 11th mark. How high would this water rise if it were poured into the empty wide cylinder?

Adding a Question:

If the administrator chooses to add a question, then a form for entering a new question, answers, reasons, and a picture are presented. There are always six available slots for answers and reasons. Note: More slots for answers can be added as needed. The administrator is asked to select the type of question for grading purposes, individual or conjunctive credit. The admin is also asked for the difficulty level and thinking type of each question.

However, the administrator may only add one question and corresponding answers at a time. Also, the administrator needs to supply the corresponding, correct answer to each question and reason.

The administrator is also responsible for providing the picture with the question. If there isn't a picture that is with the question, then the picture information is left blank. Otherwise, the administrator needs to supply the path to where the picture is stored either by typing in the full path or clicking the browse button to look for the file.


University of Montana - Higher Order Thinking Skills Test - Microsoft Internet Explorer

http://mroccs.cs.umt.edu/cgi-user/rparham/HOTS/interface.cgi?choice=add&submit=Change+Question+Database

University of Montana
Higher Order Thinking Skills Test

Adding a Question:

Enter the grading type for the question:
- ○ Individual Credit
- ○ Conjunctive Credit

Enter the level of difficulty for the question:
- ○ Easy
- ○ Medium
- ○ Hard

Enter the type of thinking involved in question:
Conservation of volume

Enter the question you want to add:

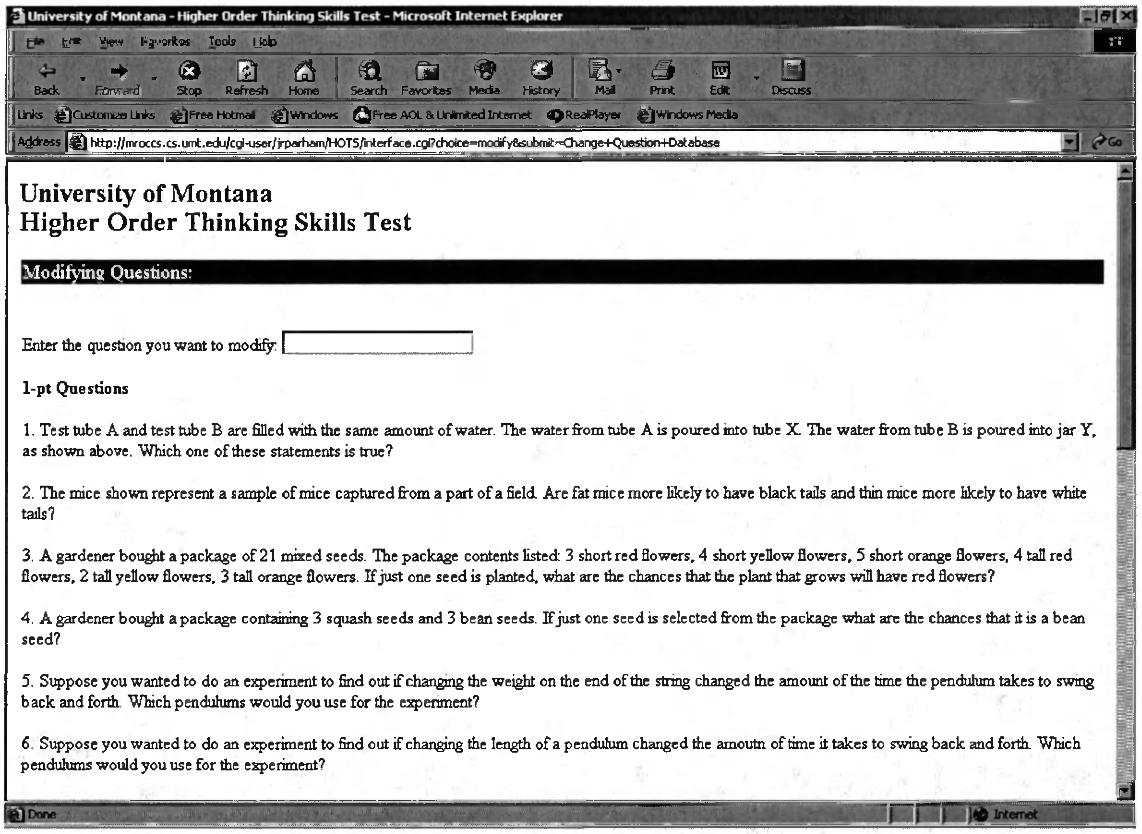
Corresponding Answers:

Modifying a Question:

Once a question has been added, the question can be modified. The question may need to be modified if there was a spelling mistake, picture error, grading type error, etc. The administrator is presented with a list of available questions from both databases that can be modified. This page is very similar to the "Delete a Question" page, however, only one question may be chosen to modify.

The question to modify must not be outside the range of questions presented to the administrator. If the question supplied by the administrator is invalid, then an invalid question page is presented.

The question to be modified is displayed to the administrator just as it appeared in the "Add a Question" page before submitting the question to the database. The administrator can modify any part of the question, and update the question in the database. If the picture of the question has not changed, then the modified picture area must be left blank.


University of Montana - Higher Order Thinking Skills Test - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites Media History Mail Print Edit Discuss
Links Customize Links Free Hotmail Windows Free AOL & Unlimited Internet RealPlayer Windows Media
Address http://mroccs.cs.umt.edu/cgi-user/jrparham/HOTS/interface.cgi?choice=modify&submit=Change+Question+Database
Go
Done
Internet

University of Montana
Higher Order Thinking Skills Test

Modifying Questions:

Enter the question you want to modify:

1-pt Questions

1. Test tube A and test tube B are filled with the same amount of water. The water from tube A is poured into tube X. The water from tube B is poured into jar Y, as shown above. Which one of these statements is true?

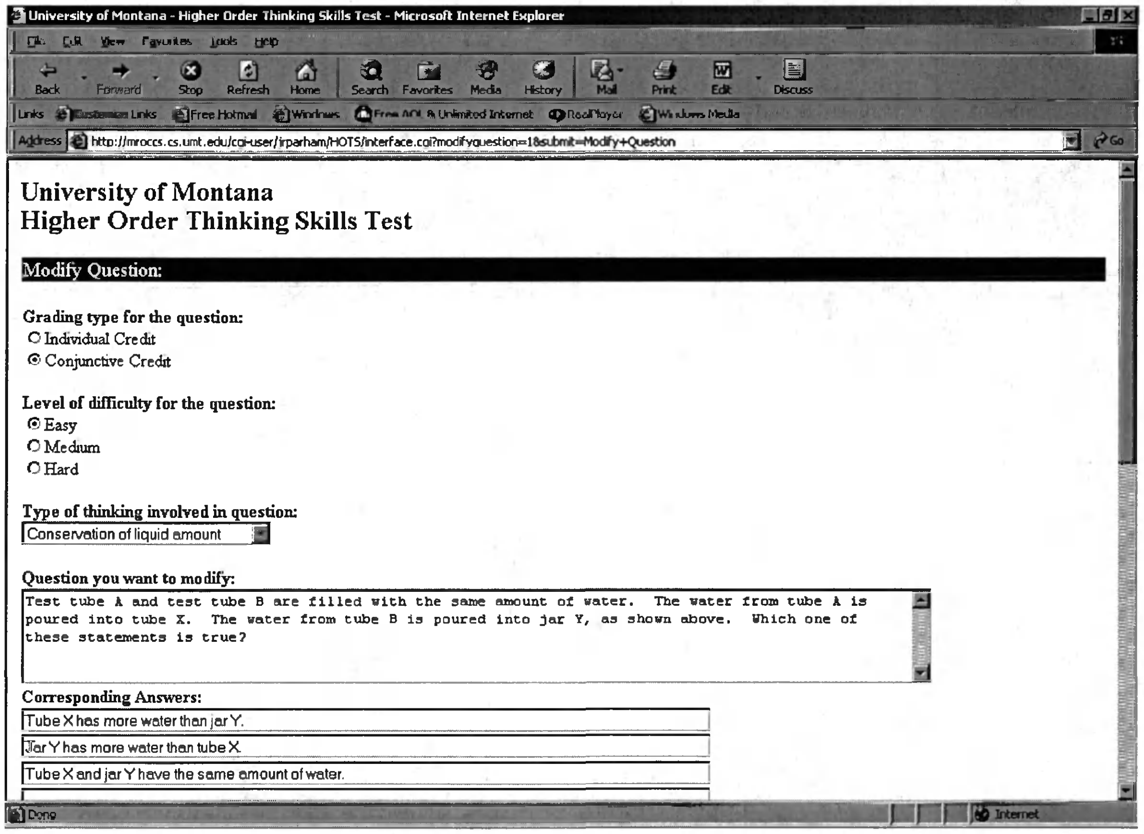
2. The mice shown represent a sample of mice captured from a part of a field. Are fat mice more likely to have black tails and thin mice more likely to have white tails?

3. A gardener bought a package of 21 mixed seeds. The package contents listed: 3 short red flowers, 4 short yellow flowers, 5 short orange flowers, 4 tall red flowers, 2 tall yellow flowers, 3 tall orange flowers. If just one seed is planted, what are the chances that the plant that grows will have red flowers?

4. A gardener bought a package containing 3 squash seeds and 3 bean seeds. If just one seed is selected from the package what are the chances that it is a bean seed?

5. Suppose you wanted to do an experiment to find out if changing the weight on the end of the string changed the amount of the time the pendulum takes to swing back and forth. Which pendulums would you use for the experiment?

6. Suppose you wanted to do an experiment to find out if changing the length of a pendulum changed the amoutn of time it takes to swing back and forth. Which pendulums would you use for the experiment?


University of Montana - Higher Order Thinking Skills Test - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites Media History Mail Print Edit Discuss
Links Customize Links Free Hotmail Windows Free AOL & Unlimited Internet RealPlayer Windows Media
Address http://mroccs.cs.umt.edu/cgi-user/jrparham/HOTS/interface.cgi?modifyquestion=1&submit=Modify+Question
Go
Done
Internet

University of Montana
Higher Order Thinking Skills Test

Modify Question:

Grading type for the question:
○ Individual Credit
[✓ Conjunctive Credit]

Level of difficulty for the question:
[✓ Easy]
○ Medium
○ Hard

Type of thinking involved in question:
Conservation of liquid amount

Question you want to modify:

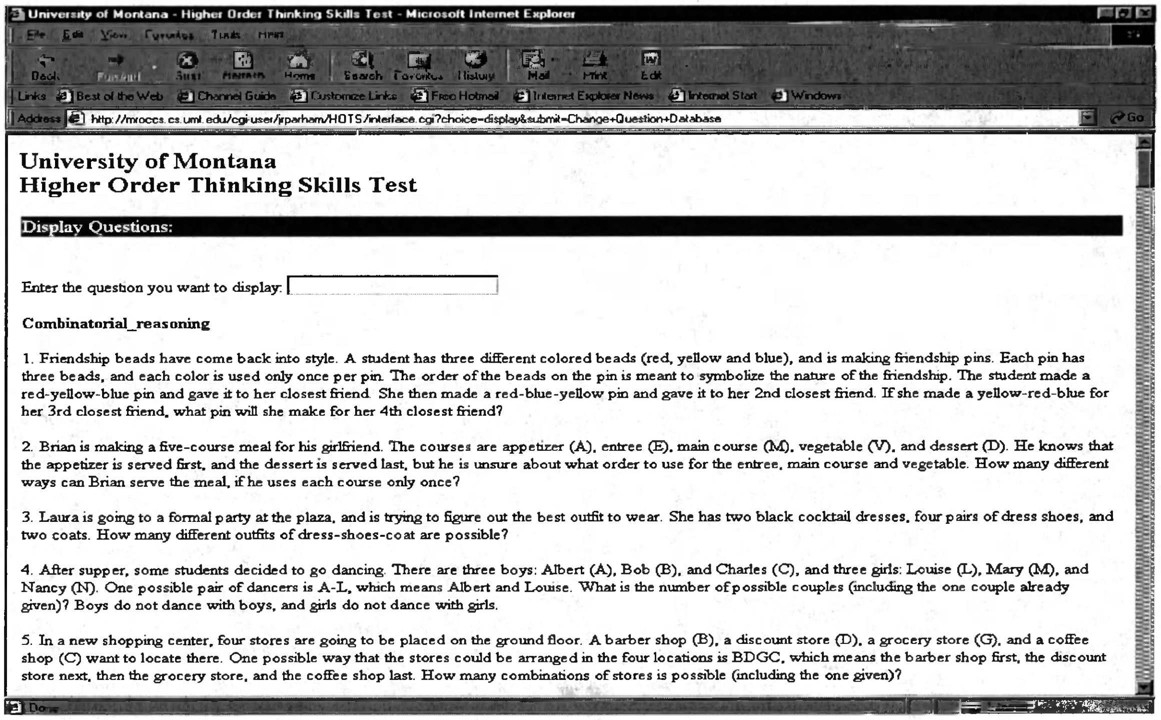
```
Test tube A and test tube B are filled with the same amount of water.  The water from tube A is
poured into tube X.  The water from tube B is poured into jar Y, as shown above.  Which one of
these statements is true?
```

Corresponding Answers:
Tube X has more water than jar Y.
Jar Y has more water than tube X.
Tube X and jar Y have the same amount of water.

Displaying a Question:

Once a question has been added, the question can be viewed. The administrator may need to view a question to check for spelling mistakes, picture errors, etc. The administrator is presented with a list of available questions from all databases that can be viewed. This page is very similar to the "Modify a Question" page.

The question to modify must not be outside the range of questions presented to the administrator. If the question supplied by the administrator is invalid, then an invalid question page is presented.


University of Montana - Higher Order Thinking Skills Test - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Links Best of the Web Channel Guide Customize Links Free Hotmail Internet Explorer News Internet Start Windows
Address http://mroccs.cs.umt.edu/cgi-user/jrparham/HOTS/interface.cgi?choice=display&submit=Change+Question+Database
Go
Done

University of Montana
Higher Order Thinking Skills Test

Display Questions:

Enter the question you want to display:

Combinatorial_reasoning

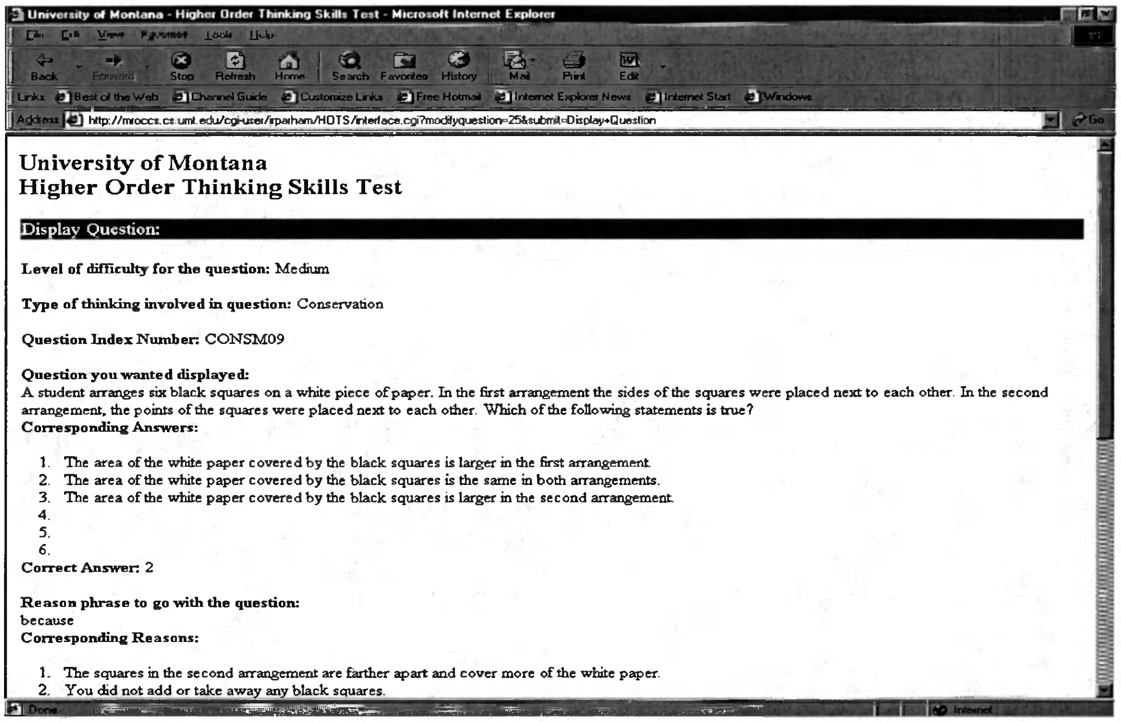
1. Friendship beads have come back into style. A student has three different colored beads (red, yellow and blue), and is making friendship pins. Each pin has three beads, and each color is used only once per pin. The order of the beads on the pin is meant to symbolize the nature of the friendship. The student made a red-yellow-blue pin and gave it to her closest friend. She then made a red-blue-yellow pin and gave it to her 2nd closest friend. If she made a yellow-red-blue for her 3rd closest friend, what pin will she make for her 4th closest friend?

2. Brian is making a five-course meal for his girlfriend. The courses are appetizer (A), entree (E), main course (M), vegetable (V), and dessert (D). He knows that the appetizer is served first, and the dessert is served last, but he is unsure about what order to use for the entree, main course and vegetable. How many different ways can Brian serve the meal, if he uses each course only once?

3. Laura is going to a formal party at the plaza, and is trying to figure out the best outfit to wear. She has two black cocktail dresses, four pairs of dress shoes, and two coats. How many different outfits of dress-shoes-coat are possible?

4. After supper, some students decided to go dancing. There are three boys: Albert (A), Bob (B), and Charles (C), and three girls: Louise (L), Mary (M), and Nancy (N). One possible pair of dancers is A-L, which means Albert and Louise. What is the number of possible couples (including the one couple already given)? Boys do not dance with boys, and girls do not dance with girls.

5. In a new shopping center, four stores are going to be placed on the ground floor. A barber shop (B), a discount store (D), a grocery store (G), and a coffee shop (C) want to locate there. One possible way that the stores could be arranged in the four locations is BDGC, which means the barber shop first, the discount store next, then the grocery store, and the coffee shop last. How many combinations of stores is possible (including the one given)?


University of Montana - Higher Order Thinking Skills Test - Microsoft Internet Explorer
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Mail
Print
Edit
Links
Best of the Web
Channel Guide
Customize Links
Free Hotmail
Internet Explorer News
Internet Start
Windows
Address
http://moccs.cs.umt.edu/cgi-user/rparham/HOTS/interface.cgi?modifyquestion=25&submit=Display+Question
Go
Done
Internet

University of Montana
Higher Order Thinking Skills Test

Display Question:

Level of difficulty for the question: Medium

Type of thinking involved in question: Conservation

Question Index Number: CONSM09

Question you wanted displayed:
A student arranges six black squares on a white piece of paper. In the first arrangement the sides of the squares were placed next to each other. In the second arrangement, the points of the squares were placed next to each other. Which of the following statements is true?
Corresponding Answers:

1. The area of the white paper covered by the black squares is larger in the first arrangement.
2. The area of the white paper covered by the black squares is the same in both arrangements.
3. The area of the white paper covered by the black squares is larger in the second arrangement.
4.
5.
6.

Correct Answer: 2

Reason phrase to go with the question:
because
Corresponding Reasons:

1. The squares in the second arrangement are farther apart and cover more of the white paper.
2. You did not add or take away any black squares.