



NATURAL SCIENCES



ZOLTÁN KÁTAI

**ALGORYTHMICS:
TECHNOLOGICALLY
AND ARTISTICALLY ENHANCED
COMPUTER SCIENCE EDUCATION**

ZOLTÁN KÁTAI

*ALGORYTHMICS:
TECHNOLOGICALLY
AND ARTISTICALLY ENHANCED
COMPUTER SCIENCE
EDUCATION*

S SAPIENTIA BOOKS



SAPIENTIA
HUNGARIAN UNIVERSITY
OF TRANSYLVANIA

***ALGORYTHMICS:
TECHNOLOGICALLY
AND ARTISTICALLY ENHANCED
COMPUTER SCIENCE EDUCATION***

ZOLTÁN KÁTAI

| Scientia Publishing House |
| Cluj-Napoca · 2021 |

SAPIENTIA BOOKS

Natural Sciences

Supported by:



Published by

Sapientia Publishing House

400112 Cluj-Napoca, Matei Corvin Street 4.

Tel./fax: +40-364-401454, e-mail: scientia@kpi.sapientia.ro

www.scientiakiado.ro

Publisher-in-Chief:

Angella Sorbán

Consultant:

Mária Csernoch (Debrecen)

Publishing coordinator:

Beáta Szabó

© Scientia, 2021

All rights reserved, including the rights for photocopying, public lecturing, radio and television broadcast, and translation of the whole work and of chapters as well.

Descrierea CIP a Bibliotecii Naționale a României

ZOLTÁN, KÁTAI

Algorithms : technologically and artistically enhanced computer science education / Zoltán Káti. - Cluj-Napoca : Scientia, 2021

Conține bibliografie

ISBN 978-606-975-044-5

004

CONTENTS

1 Introduction	17
1.1 AlgoRythmics: An award-winning project	18
1.2 The AlgoRythmics research group	19
2 Multi-sensory computer science education	21
2.1 Difficulties in teaching-learning programming.	21
2.2 Brain-based (multi-sensory) learning	22
2.2.1 Memory and multi-sensory learning.	24
2.3 Technologically enhanced multi-sensory learning.	25
2.3.1 Hybrid/blended learning	25
2.4 Multimedia and multi-sensory learning at all levels	26
2.5 Multi-sensory learning through arts	27
2.5.1 Combining science education with arts	28
2.5.1.1 Science education on stage	28
2.5.1.2 Arts in science classrooms	29
2.6 On the role of senses in education	30
3 Seeing, hearing, and touching computer algorithms (Study 1).	31
3.1 Anatomy of simple algorithms	31
3.2 Software tool	33
3.3 Suggested syllabus	35
3.4 The experiment	38
3.4.1 Results and discussion	39
4 Playing recursive scenarios (Study 2)	43
4.1 Teaching/learning recursion	43
4.1.1 How to teach recursive procedure design?	43
4.1.2 How to teach recursive function design?	45
4.2 How can the presented methods be improved by kinaesthesia?	46
4.3 How can the presented methods be improved by audio-visual elements?	48
4.4 Software tool and suggested syllabus	50
4.5 The experiment	52
5 Dancing sorting algorithms (Study 3).	55
5.1 Software tool and suggested syllabus	56
5.2 The experiment	57

6 Some conclusions on our multi-sensory computer science education research	59
7 The AlgoRythmics learning environment	61
7.1 The AlgoRythmics dance performance collection	62
7.1.1 Evidence of impact	65
7.2 AlgoRythmics animations	68
8 AlgoRythmics: Science and art without ethnic borders (Study 4)	73
8.1 Artistically enhanced multicultural education	73
8.2 Intercultural computer science education in Transylvania	76
8.2.1 Results	78
8.3 Conclusions	80
9 Teaching “not blind learners” to program “blind computers” (Study 5)	83
9.1 Theoretical background	83
9.2 Student-orchestrated computer algorithms	85
9.3 Selective hiding for improved algorithm visualization	86
9.4 The experiment	87
9.4.1 Results and discussion	88
9.5 Conclusions	90
10 Promoting algorithmic/computational thinking of both sciences- and humanities-oriented learners (studies 6 and 7)	93
10.1 The “two cultures”.	94
10.2 Promoting algorithmic/computational thinking	96
10.3 The motivational perspective	98
10.4 The experiment	100
10.5 Results regarding students’ performance	103
10.6 Results regarding the motivational perspective	105
10.6.1 Limitations	110
10.7 Conclusions	110
11 Multidimensional expansion of the AlgoRythmics environment	115
11.1 Expanding the AlgoRythmics collection.	115
11.1.1 From 1D view to 2D view	115
11.1.2 Including new dance styles.	117
11.1.3 Moving on to a new algorithm family.	117
11.1.4 Feedback.	121
11.2 From dance to code	123
11.2.1 Learning steps	124

11.2.2 Courses	125
11.2.3 Levels of interactivity	126
11.3 Promoting computational thinking in the extended AlgoRythmics environment	126
11.3.1 Shifting to blended learning	127
11.3.2 Exploring searching strategies from an algorithm complexity perspective	128
11.3.3 Exploring sorting strategies from an algorithm complexity perspective	129
11.3.4 Exploring backtracking strategies from an algorithm complexity perspective	131
11.3.5 Basic characteristics of algorithms: Generality	131
11.3.6 Computer algorithm “optimization”	132
12 Algorithm visualization environments: Can an optimal interactivity level be established? (Study 8)	135
12.1 Different levels of engagement with algorithm visualizations	135
12.2 Different levels of engagement in the AlgoRythmics environment	137
12.3 The experiment	139
12.4 Results and discussion	140
12.4.1 Results grouped by prior programming experience	142
12.4.2 Relations between the level of interactivity and the nature of acquired knowledge	144
12.4.3 Results grouped by gender	146
12.4.4 Most preferred course variant	147
12.4.5 Limitations	149
12.5 Conclusions	149
13 Ongoing research	151
13.1 Schematic versus human-movement-enriched realistic algorithm visualization	151
13.1.1 Abstract animation versus dance choreography (Study 9)	153
13.1.2 Combining schematic and realistic visualizations in the AlgoRythmics environment (Study 10)	154
13.2 Improving AlgoRythmics teaching-learning environment by asking questions (Study 11)	155
13.3 Investigating young school students’ computational thinking ability across grade levels (Study 12)	156
14 AlgoRythmics: Past, present, and future	159
14.1 The renewed learning environment	159
14.2 Research in the AlgoRythmics environment	161

14.3 Plans for the near future	161
14.4 Final conclusion.	161
References	163
Appendix	183
About the Author	193

TARTALOM

1. Bevezető	17
1.1. AlgoRhythmic: egy díjazott projekt	18
1.2. Az AlgoRhythmic kutatócsoport	19
2. Többérzékszerves informatikaoktatás	21
2.1. Kihívások a programozásoktatásban	21
2.2. Az emberi agy és a tanulás	22
2.2.1. Az emlékezet és a többérzékszerves tanulás	24
2.3. Technológia a többérzékszerves tanulás szolgálatában	25
2.3.1. Vegyes/kombinált tanulás	25
2.4. Multimédia és a többérzékszerves tanulás	26
2.5. Többérzékszerves tanulás művészetközeli elemek révén	27
2.5.1. Tudomány és művészet: egy erős kombináció	28
2.6. Az érzékszervek szerepe az oktatásban	30
3. Látni, hallani és tapintani a számítógépes algoritmusokat (1. tanulmány) ..	31
3.1. Az egyszerű algoritmusok anatómiája	31
3.2. A szoftvereszköz	33
3.3. Javasolt tanmenet	35
3.4. A kísérlet	38
3.4.1. Eredmények és következtetések	39
4. Rekurzív algoritmusok lejátszódva (2. tanulmány)	43
4.1. Rekurzív algoritmusok oktatása	43
4.1.1. Hogyan tanítsunk rekurzív eljárásokat?	43
4.1.2. Hogyan tanítsunk rekurzív függvényeket?	45
4.2. A kinezetikus érzékelés szerepe a programozásoktatásban	46
4.3. Audiovizuális elemekkel dúsított módszerek	48
4.4. Szoftvereszköz és javasolt tanmenet	50
4.5. A kísérlet	52
5. Eltáncolt algoritmusok (3. tanulmány)	55
5.1. Szoftvereszköz és javasolt tanmenet	56
5.2. A kísérlet	57
6. Kutatási eredmények a többérzékszerves programozásoktatás kapcsán ...	59

7. Az AlgoRythmics tanulási környezet	61
7.1. Az AlgoRythmics tánckoreográfia-gyűjtemény	62
7.1.1. AlgoRythmics visszhang	65
7.2. AlgoRythmics animációk	68
8. AlgoRythmics: tudomány és művészet etnikai határok nélkül	
(4. tanulmány)	73
8.1. Művészetközeli elemekkel gazdagított multikulturális informatikaoktatás	73
8.2. Interkulturális programozásoktatás Erdélyben	76
8.2.1. Eredmények	78
8.3. Következtetések	80
9. Amikor látó diákok programoznak vak számítógépeket	
(5. tanulmány)	83
9.1. Elméleti háttér	83
9.2. Levezényelt algoritmusok	85
9.3. Az elrejtés mint módszer a hatékonyabb algoritmusvizualizációhoz	86
9.4. A kísérlet	87
9.4.1. Eredmények	88
9.5. Következtetések	90
10. Algoritmikus és számítógépes gondolkodás:	
humán vs. reál szakos diákok (6. és 7. tanulmány)	93
10.1. A „két kultúra”	94
10.2. Hogyan fejleszthető az algoritmikus, illetve számítógépes gondolkodás?	96
10.3. A motiváció szerepe	98
10.4. A kísérlet	100
10.5. Eredmények a diákok teljesítménye szempontjából	103
10.6. Eredmények a diákok motivációja szempontjából	105
10.6.1. A kutatás korlátai	110
10.7. Következtetések	110
11. A megújult AlgoRythmics környezet	115
11.1. A kibővült AlgoRythmics kollekció	115
11.1.1. Algoritmusvizualizációk 2D-ben	115
11.1.2. Új táncstílusok	117
11.1.3. Új algoritmuscsaládok	117
11.1.4. Felerősödött AlgoRythmics visszhang	121
11.2. A táncról a kódig	123

11.2.1. Tanulási lépések	124
11.2.2. AlgoRhythmic tanmenetek	125
11.2.3. Interaktivitási szintek	126
11.3. Számítógépes gondolkodás fejlesztése a kibővült AlgoRhythmic környezetben	126
11.3.1. Elmozdulás a vegyes oktatás fele	127
11.3.2. Keresési stratégiák algoritmusbonyolultsági szempontból	128
11.3.3. Rendezési stratégiák algoritmusbonyolultsági szempontból	129
11.3.4. Visszalépéses keresés: algoritmusbonyolultsági szempontok	131
11.3.5. Az algoritmusok általános jellegének kiemelése	131
11.3.6. Algoritmushatékonyaság	132
12. Algoritmusvizualizációs környezetek: létezik optimális interaktivitási szint? (8. tanulmány)	135
12.1. Interaktivitási szintek algoritmusvizualizációs környezetekben	135
12.2. Interaktivitási szintek az AlgoRhythmic környezetekben	137
12.3. A kísérlet	139
12.4. Eredmények	140
12.4.1. Következtetések az előzetes programozói tapasztalat szempontjából	142
12.4.2. Kapcsolat az interaktivitási szint és a megszerzett tudás jellege között	144
12.4.3. Következtetések a diákok neme szempontjából	146
12.4.4. A legkedveltebb tanmenet	147
12.4.5. A kutatás korlátai	149
12.5. Következtetések	149
13. Zajló kutatásaink	151
13.1. Az emberi mozgás effektus kamatoztatása az algoritmusvizualizációban	151
13.1.1. Absztrakt animációk versus realisztikus tánckoreográfiák (9. tanulmány)	153
13.1.2. Sematikus és realisztikus ábrázolások az AlgoRhythmic környezetben (10. tanulmány)	154
13.2. Kérdéssorozatok társítása az AlgoRhythmic környezethez (11. tanulmány)	155
13.3. Elemista és gimnazista tanulók számítógépes gondolkodásának vizsgálata (12. tanulmány)	156
14. AlgoRhythmic: múlt, jelen és jövő	159
14.1. Megújult tanulási környezet	159

14.2. Oktatástudományi-kutatások az AlgoRythmics környezetben	161
14.3. Jövőbeli tervek	161
14.4. Végző következtetések	161
Irodalomjegyzék	163
Függelék	183
Kivonat	189
A szerzőről	193

CONȚINUT

1. Introducere	17
1.1. AlgoRhythmic: un proiect premiat	18
1.2. Grupul de cercetare AlgoRhythmic	19
2. Predarea informaticii: metode multi-senzoriale	21
2.1. Dificultăți în predarea-învățarea programării calculatoarelor	21
2.2. Învățare multi-senzorială	22
2.3. Învățare multi-senzorială asistată de calculator	25
2.4. Învățare multimedia și multi-senzorială	26
2.5. Promovarea învățării multi-senzoriale prin elemente artistice	27
2.6. Rolul simțurilor în învățare	36
3. Antrenarea văzului, auzului și a simțului tactil în învățarea algoritmilor (Studiul 1)	31
3.1. Anatomia algoritmilor simpli	31
3.2. Aplicația software	33
3.3. Planul de învățământ sugerat	35
3.4. Experiment didactic	38
4. Role-playing: scenarii recursive (Studiul 2)	43
4.1. Predarea-învățarea recursivității	43
4.2. Îmbunătățirea metodei prin antrenare simțului chinestezic	46
4.3. Îmbunătățirea metodei prin elemente audiovizuale	48
4.4. Aplicația software și planul de învățământ sugerat	56
4.5. Experiment didactic	52
5. Algoritmi de sortare ilustrați prin coreografii de dans (Studiul 3)	55
5.1. Aplicația software și planul de învățământ sugerat	56
5.2. Experiment didactic	57
6. Concluzii privind metodele multi-senzoriale în domeniul predării-învățării programării calculatoarelor	59
7. Mediul de învățare AlgoRhythmic	61
7.1. Colecția de coreografii AlgoRhythmic	62
7.2. Animații AlgoRhythmic	68

8. AlgoRythmics: știință și artă fără granițe etnice (Studiul 4)	73
8.1. Rolul elementelor artistice în educația multiculturală	73
8.2. Predarea și învățarea informaticii în Transilvania: metode cu caracter intercultural	76
8.3. Concluzii	80
9. Studenți programând “calculatorul nevăzător” (Studiul 5)	83
9.1. Bază teoretică	83
9.2. Orchestrarea algoritmilor	85
9.3. “Ascundere selectivă” pentru o vizualizare mai eficientă	86
9.4. Experiment didactic	87
9.5. Concluzii	90
10. Gândire algoritmică/computațională la studenții cu profil uman și real (Studiile 6 și 7)	93
10.1. “Cele două culturi”	94
10.2. Promovarea gândirii algoritmice/computaționale	96
10.3. O perspectivă motivațională	98
10.4. Experiment didactic	100
10.5. Rezultate privind performanța studenților	103
10.6. Rezultate privind aspectele motivaționale	105
10.7. Concluzii	110
11. Expansiunea multidimensională a mediului de învățare AlgoRythmics	115
11.1. Colecția AlgoRythmics îmbogățită	115
11.2. De la dans până la cod	123
11.3. Promovarea gândirii computaționale în mediul AlgoRythmics extins	126
12. Medii de vizualizarea a algoritmilor: există nivel de interactivitate optim? (Studiul 8)	135
12.1. Rolul interactivității în mediile de vizualizare a algoritmilor	135
12.2. Nivele de interactivitate cu aplicația AlgoRythmics	137
12.3. Experiment didactic	139
12.4. Rezultate	140
12.5. Concluzii	149
13. Cercetări în curs	151
13.1. Algoritmi: vizualizări schematice versus vizualizări realiste (Studiile 9 și 10)	151

13.2. Eficientizarea învățării în mediului AlgoRhythmic prin arta întrebării (Studiul 11)	155
13.3. Investigarea gândirii computaționale la elevii din învățământul elementar, gimnazial (Studiul 12)	156
14. AlgoRhythmic: trecutul, prezentul și viitorul	159
14.1. Aplicația reînnoită	159
14.2. Cercetare în mediul de învățare AlgoRhythmic	161
14.3. Planuri pentru viitor	161
14.4. Concluzii	161
Bibliografie	163
Apendice	185
Rezumat	189
Despre autor	193

1 INTRODUCTION

A major responsibility of educational systems in the 21st century is to prepare future generations for the challenges involved with the increasing computerization of our everyday lives and to meet the demands of one of the fastest-growing job markets: computing (Grover & Pea, 2013; US-BLS, 2020). In line with this, in 2011, the *Future Work Skills* report of the Institute for the Future included computational thinking (CT) among the 10 top skills that will be needed for success in 2020 (Davies, Fidler, & Gorbis, 2011).

Shute, Sun, and Asbell-Clarke (2017) draw an interesting analogy between reading-writing and CT. In the mediaeval period, only select groups of people could read and write, but as the world evolved increasingly more people needed these skills. Similarly, in the digital world of the 21st century, everyone should acquire CT, not only programmers. CT has become a very hot topic in educational research and practice after Jeanette Wing published an influential article in this topic in 2006. According to Wing, CT is merely thinking like a computer scientist when approaching a problem and in solving it. As CT grew in popularity, computing education also received more and more attention. In the UK and the US, these trends are evident from initiatives such as *Computing at School* and *Computer Science for ALL*.

The most cited definition of CT emphasizes that CT is a thinking process where “solutions are represented in a form that can be effectively carried out by an information-processing agent” (Wing 2010). To better understand the nature of this concept, researchers have tried to identify its roots within the framework of modern educational culture.

The term of CT stems back to the constructionist work of Seymour Papert (1980, 1996). According to Spangenberg and Brynskov (2018), Papert’s work is a good starting point for talking about computing education from the perspective of CT. Papert formulated three main principles: (1) the power principle emphasizes that the natural mode of acquiring knowledge is through use, which will progressively lead to the deepening of one’s understanding; (2) the thingness principle is concerned with making abstract ideas concrete through a meaningful representation; (3) the dynamics before statics principle is closely related to the medium used for teaching.

With regard to the expression of “can be effectively carried out by an information-processing agent”, Benedict du Boulay is recognized to be the first who introduced the concept of the notional machine. He used this term in the context of teaching novices how to program: “The notional machine is an idealized, conceptual computer whose properties are implied by the constructs in the programming language employed” (du Boulay, O’Shea, & Monk, 1981).

Wing’s definition of CT has recently been a target for critiques. For example, Denning (2017) distinguishes between Traditional CT (pre-2006) and New CT

(post-2006): “programming ability produces CT” versus “learning certain concepts produces programming ability”. He promotes Aho’s (2012) definition of CT: the thought process involved in formulating problems so that “their solutions can be represented as computational steps and algorithms”. Accordingly, Denning emphasizes that algorithms are central to CT, and, consequently, CT and algorithmic thinking (AT) are strongly related concepts. He also underlines that algorithms, in the context of CT, must control some computational model.

The goal of our beloved AlgoRhythmics project is to promote computing education for all by taking into account the above highlighted elements from CT definitions. For this purpose, we created an engaging algorithm visualization environment. The environment is built around a collection of interactive dynamic visualizations illustrating basic computer algorithms.

Making computing education attractive for different categories of learners (including K–12 learners and non-CS majors) is a challenging initiative. According to Guzdial (2010), a possible approach might be contextualization. Since developing differentiated teaching-learning strategies may involve substantial additional costs, some scholars have tried to find a context that is appealing to most students. A promising candidate for this “common denominator role” could be arts. The AlgoRhythmics learning environment has been designed along this approach. Since music and dance are relatively close to most young people, this environment visualizes searching and sorting algorithms by professional dance choreographies (folkdance, flamenco, ballet).

As an introduction and to arouse interest, perhaps, that is enough. What is this book about? About the AlgoRhythmics universe. Of course, we did not dream of a complex teaching-learning tool and the attached didactical methods overnight. The AlgoRhythmics project has its own particular history. Through this book, we invite the reader to accompany us as we virtually relive the AlgoRhythmics adventure.

1.1 AlgoRhythmics: An award-winning project

A 2013 report by the joint Informatics Europe & ACM Europe Working Group on Informatics Education (IE & ACM, 2013) states that for a nation or group of nations to compete in the race for technological innovation, the general population must understand the basics of informatics: the science behind information technology (IT). To be competitive in the 21st century’s job market, students must understand the key concepts of informatics. The report describes CT as an important ability that all people should possess. The working group emphasizes that informatics-based concepts, abilities, and skills are teachable and must be included in the primary and particularly in the secondary school curriculum.

Accordingly, the “2013 Best Practices in Education Award” (organized by Informatics Europe) was devoted to initiatives promoting *Informatics Education in*

Primary and Secondary Schools. The winners were presented in a special ceremony held during the ECSS 2013 in Amsterdam, The Netherlands. Two teams from Eastern/Central Europe (Romania: Sapientia Hungarian University of Transylvania; Poland: Warsaw School of Computer Science) shared that year's award. The official website of Informatics Europe states:

The evaluation committee praised the originality of the proposal by Zoltán Kátai, László Tóth, and Alpár Károly Adorjáni: Multi-Sensory Informatics Education. Mixing algorithm-learning with sensory experience is a very innovative teaching experiment. The key concept of this proposal is Computer Science (CS) education for all, using a creative approach. The committee was impressed and appreciated this approach of abstracting away almost all details that might hinder understanding the idea or principle of an algorithm or a paradigm. The enactments thus not only can be used flexibly in teaching environments irrespective of a particular programming or spoken-language but can be used as a starting point for the teacher to drill down into more technical concepts. Another particularity of the project is its inter-cultural character – sorting algorithms illustrated by Central European folk dancing (Informatics Europe, 2013).

In the years since 2013, the AlgoRythmics project has expanded in a number of areas. In this book, we provide a brief description of our fifteen-year research on the topic of technologically and artistically enhanced multi-sensory computer-programming education. This overview is based on the following research papers:

- On the role of senses in education (Kátai, Juhász, & Adorjáni, 2008);
- Technologically and artistically enhanced multi-sensory computer-programming education (Kátai & Tóth, 2010);
- Multi-sensory method for teaching-learning recursion (Kátai, 2011);
- Selective hiding for improved algorithmic visualization (Kátai, 2014a);
- Intercultural Computer Science education (Kátai, 2014b);
- The challenge of promoting algorithmic thinking of both sciences- and humanities-oriented learners (Kátai, 2015);
- Promoting computational thinking of both sciences- and humanities-oriented students: An instructional and motivational design perspective (Kátai, 2020);
- Algorithm visualization environments: Degree of interactivity as an influence on student learning (Osztíán, Kátai, & Osztíán, 2020).

1.2 The AlgoRythmics research group

The AlgoRythmics project started during the 2003–2007 period at Sapientia Hungarian University of Transylvania. At that time, the author (Zoltán Kátai) was PhD student at the University of Debrecen, and one of the topics he addressed was the multi-sensory approach of CS education. The first investigation that can be linked to the project (included in the author's PhD dissertation too) focused on

the role of senses in education. The partner involved in this study was an undergraduate student, Alpár Károly Adorjáni. He developed the software tool (Code Buherator) which allowed the combined involvement of sight, hearing, and touch in the teaching-learning process of computer algorithms. Afterwards, we added the kinaesthetic sense too. In *chapters 3 and 4*, we detail the methods we developed at that stage of the project.

In the coming years, another undergraduate student was invited to participate in the project, László Tóth. He contributes to the involvement of dance in our multi-sensory computer-programming education programme (*Chapter 5*). As a next step, the research group initiated a collaboration with a professional folk dance institution (*Maros Művészegyüttes*), and six folk dance choreographies were created with the aim of illustrating sorting algorithms. These videos were posted on the AlgoRhythmic YouTube channel on 2011 (Kátai & Tóth, 2011). László Tóth developed the first version of the AlgoRhythmic web application, which associates interactive computer animations with the algorithmic dance performances. This learning environment (detailed in *Chapter 7*) provided the framework for the research studies presented in *chapters 8 to 10*.

In 2016, two new colleagues joined the group, Erika Osztián and Géza Károly Vekov. They gave the project a new impetus (see *Chapter 11*). Four new dance choreographies were added to the AlgoRhythmic collection (Kátai, Osztián, Osztián, & Vekov, 2018). We extended our repertoire with new algorithms and new dance styles (flamenco in collaboration with the András Lóránt Company; ballet in collaboration with the Cluj-Napoca Hungarian State Opera). The project entered a new stage when we decided to redesign the AlgoRhythmic web application (Kátai, Osztián, Osztián, Nagy, & Cosma, 2020). Three undergraduate students contributed to this: Pálma Rozália Osztián, Eszter Jáhel Nagy, and Cristian Sebastian Cosma. Their work was technically supervised by Csaba Tekse from Lateral Company (a design and development studio). Pálma Rozália Osztián remained a member of the research team even after graduating. *Chapters 12 and 13* report on the recent studies that were implemented, mostly in the renewed AlgoRhythmic environment.

In the first phase of the project, we focused on enhancing CS education. The subjects for the studies from this period were CS students. In *Chapter 2*, we present the theoretical background for these investigations. *Chapter 6* includes some conclusions based on the findings of our first three research studies. In the last years, we extended our research interest to other categories of learners too: humanities-oriented students and elementary and gymnasium-level learners. Because of the diversity of studies 4–8, the related literature reviews and conclusions have been included in the corresponding chapters. The *last chapter* offers a brief overview of the AlgoRhythmic project and mentions some of our future plans.

2 MULTI-SENSORY COMPUTER SCIENCE EDUCATION

During our first three research studies, we focused on supporting CS education based on the principles of multi-sensory learning. In this chapter, we analyse why teaching-learning computer programming is a challenging task and why multi-sensory approaches could enhance this educational process. The methods and instruments we designed cover the following areas: loop structures, recursive algorithms, sorting strategies.

2.1 Difficulties in teaching-learning programming

Since the early days of programming education, teachers have signalled problems regarding students' programming abilities. Researchers (cognitive scientists, learning theorists, computer scientists, etc.) have identified specific difficulties related to learning to program (Mead et al., 2006). For example, du Boulay (1986) focused on identifying problematic areas and common mistakes made in them. According to Spohrer and Soloway (1986), Winslow (1996), and Soloway, Bonar, and Ehrlich (1983), most students have problems in combining algorithmic structures into programs. Navrat (1994) emphasizes the abstractness of the programming process as a possible factor contributing to students' difficulties in learning to program. The common (disappointing) conclusions of several studies in the early 2000s were: students cannot program, trace programs, or design programs at acceptable levels (McCracken et al., 2001; Lister et al., 2004; Eckerdal, McCartney, Moström, Ratcliffe, & Zander, 2006). Another conclusion of that research period was that the problem is both long-standing and has an international character.

Research on learning scientific concepts also yields insights into why understanding complex information is difficult. Many scientific domains (also including mathematics) deal with abstract concepts that students have difficulty comprehending. Mastery of these concepts requires that students build flexible and runnable mental models. Frequently, the scientific models describe phenomena for which students have no real-life referents and incorporate invisible factors and abstractions. This is particularly true in the case of learning algorithms, which is also characterized by a high-level abstractness (Dede, Salzman, Loftin, & Sprague, 1999). The following chain of ideas confirms the multiple abstract character of programming: the programming language itself can be considered as a first-level abstraction, the computer program will be the second abstraction level, and the

algorithm behind the program may be the third-level abstraction. In addition, the generally applied problem-solving process (1. abstraction, 2. decomposition, 3. transformation into sub-solutions, 4. recombination into a working program, 5. evaluation) also starts with abstracting the problem from its description.

Therefore, an important question we have addressed is the following: How can CS teachers handle the problem of the abstractness of the programming process? The high-level abstractness itself suggests that the effectiveness of this kind of educational processes can be increased by a multiple-senses approach. A relevant example in this sense is the success of the *Making Math Real* curriculum (Berg & Knop, 2008). The *Making Math Real: Connecting Research to Practice – A Comprehensive Multisensory Structured Methodology in Mathematics K–12* workshop reviewed the work of Giedd, Sowell, Deheane, Butterworth, Geary, and others in the areas of neuroscience and cognitive science, combined with the work of Miller, Mercer, Tomey, Marolda, Orton-Gillingham, and others for the connections to the cognitive benefits of multi-sensory structured methods. Their conclusion is that these results can be considered as a research basis for the multi-sensory structured teaching methodologies.

Since students' difficulties in learning scientific concepts, mathematics, and computer algorithms are closely related, the research referred above suggests that multi-sensory approach can be efficient in the case of algorithm design too. In the following, for further support, we detail our literature review in the field of multi-sensory education.

2.2 Brain-based (multi-sensory) learning

Revolutionary discoveries in neuroscience and important developments in cognitive psychology have resulted in new ways of thinking about the relationship between senses and learning. It is more and more evident that our brain is organized to elaborate information, coming from the different sensory channels, cooperatively, in order to have a complete vision of reality (Voto, Viñas, & D'Auria, 2005). Although much traditional sensory research has studied each sensory modality separately, there has been a recent surge of interest in causal interplay between different senses (Driver & Noesselt, 2008).

Everything we know we have learned by using our senses. Each sense, either singularly or in various combinations, provides a pathway to learning. While each sense is important in itself, our senses are designed to function in harmony. Kinaesthesia has been defined as “the feedback mechanism of the nervous system which conveys information between the mind and the body” and what coordinates “our senses of hearing, sight, and touch; our faculties of knowing and reasoning; our ability to feel and to act on our feelings” (DSA, 2020; TPUB, 2020).

Traditionally, elements of perception, such as vision, hearing, smell, taste, and touch, have been viewed as additive, separate, and independent processes. However, exciting discoveries in neuroscience have disproved this theory. Researchers have identified multisensory interactions both in the case of perceptual tasks and settings and throughout processing. Multi-sensory interactions have been localized in the early sensory, association, and other cortical areas, including feed-forward and feed-back pathways (Stein & Meredith, 1993; Shimojo & Shams, 2001; Falchier, Clavagnier, Barone, & Kennedy, 2002; Schroeder & Foxe, 2002; Calvert, Spence, & Stein, 2004; Foxe & Schroeder, 2005; Ghazanfar & Schroeder, 2006; Driver & Noesselt, 2008).

Findings in brain research have demonstrated that different object characteristics are processed in different visual areas. Techniques that allow simultaneous recordings of multi-neuronal activity revealed that any particular object within our visual field is represented by the firing of a set of neurons. Bongard, Ferrandez, and Fernandez (2009) describe the neuron activity during visual information processing as a neural concert of the visual orchestra. This metaphor can be extended to other senses as well. For example, like vision, haptic processing pathways are also organized into a hierarchy of processing stages, with different stages represented by different brain areas (James, Kim, & Fisher, 2007). Additionally, James et al. refer points of neural convergence to vision and haptics. On the other hand, Overy and Turne (2009), after they had reviewed the related literature, concluded: What is most clear from this collection of papers is that the neural bases of rhythm and movement are fundamentally connected and distributed across a wide range of brain regions.

Other researchers have also identified convergent neural pathways onto multi-sensory neurons (Stein & Meredith, 1993) that may provide the substrate for multi-sensory binding (Meredith, 2002). A typical characteristic of multi-sensory neurons are that they fire only when more than one sensory modality is activated (Kavenaugh, 1991; Shaywitz, 2003; van Wassenhove, Grant, & Poeppel, 2005); they are characterized by enhanced response (supra-additivity) to the presentation of co-occurring events. Accordingly, we think that it would be reasonable to extend the visual orchestra metaphor to multi-sensory orchestra. In such an orchestra, multi-sensory neurons use multi-instruments.

The left brain and right brain expressions are used to describe the specialized functions of the two hemispheres of the human brain. For example, experiments applying neuroimaging technologies showed that activities involving numbers, logic, sequential tasks, and in general analysis are more closely associated with the left side of the brain (“academic brain”). Then again, activities involving music, imagination, colours, or creative expression are more active in the right hemisphere (“artistic brain”). While understanding the brain’s hemispheres is undoubtedly relevant to education, children cannot be categorized as exclusively left-brained or right-brained learners. Some research in this field revealed that

in a normal brain the two hemispheres operate together. In harmony with this reality, educational researchers showed that a balanced involvement of both sides of the brain in the classroom can significantly improve the teaching-learning process (Bransford, Brown, & Cocking, 1999; Eisenhower SCIMAST, 1997).

The work of Howard Gardner has also revealed that each man has a mixture of different ways of learning. In his first book, *Frames of Mind*, Gardner (1993) identified seven “intelligences”. Subsequently, an eighth and a ninth intelligence were added to the original list. This list includes, among others, the musical, the bodily-kinaesthetic and the logical-mathematical intelligences. Gardner calls attention that people are born with all intelligences but usually only one or two are completely developed in any individual. One of the important messages of Gardner’s work for all teachers is that students need to learn in various ways, not only in their obvious and most natural way. For example, teachers should not permit for their visual or logical learners to rely only on their most comfortable intelligence (Eisenhower SCIMAST, 1998).

2.2.1 Memory and multi-sensory learning

Another vital element of the learning process is memorizing. If we do not retain the learned matters, how shall we be able to utilize our knowledge? “Tantum scimus quantum memoria tenemus.” It has been estimated that people retain only 10% of what they read, 20% of what they hear, and 30% of what they see. However, a striking improvement takes place in retention if the above-mentioned senses are combined (TPUB, 2020). The same evaluations tell us that when someone hears and sees the subject at the same time retention jumps to 50%. If questions that stimulate thinking are used as a background for the eyesight and sound, retention level can be pushed close to 70%. If along with procedural steps and principles, the students are asked to use all their senses in skill training, then their retention can be increased to as much as 90% (TPUB, 2020). All this implies a fair degree of mastery of teaching and learning.

The more senses are used in presenting or exploring new material, the greater the possibility is that this will be recalled by students in the future. This can be explained by the fact that there will be more pathways of locating the stored information. Furthermore, there are people who prefer auditory learning style, others favour visual ones, and others have strengths in receiving information through their kinaesthetic senses (OEF, 2001). Consequently, a multiple senses approach of education will provide equality of chances for each student.

The path from sensation to memory is a complicated process. The senses are bombarded by stimuli that must be encoded into meaningful patterns (in the working memory) and then sorted in the long-term memory (Mead et al., 2006). According to the dual coding theory, sensations are handled by two different subsystems. Verbal input is handled by a subsystem specialized in language,

while non-verbal input by a subsystem specialized in images or sensations. These images can be visual, auditory, or kinaesthetic. During data transferring (from sensation to long-term memory), the two subsystems interact. Researches revealed that memories of images are more easily recalled, while verbal memories are more easily applied, synthesized, and transferred (SDSU-ET, 2008).

Memory and learning also depend on types of sensation. If one subsystem must attend to two sensory types, information can be lost, causing inefficient memorization. If each subsystem attends to information from different sensory types, the inverse phenomenon takes place, namely, attention and memory are reinforced (SDSU-ET, 2008).

2.3 Technologically enhanced multi-sensory learning

Montessori initiated the multi-sensory learning movement about 90 years ago. In recent decades, technology integration in education has opened up new vistas for researchers and teachers who are interested in multi-sensory teaching-learning methods. Reflecting on terms like multimedia and multi-sensory, we understand that the nearly one-hundred-year-old multi-sensory movement has entered a new dynamic era.

2.3.1 Hybrid/blended learning

Digital elements have moved multi-sensory learning closer to other modern educational concepts such as hybrid and blended learning. Hybrid education combines traditional face-to-face instruction with online technologies (Swenson & Evans, 2003). While most of the research failed to find statistically significant differences between the efficacy of the online and face-to-face learning (Coates, Humphreys, Kane, & Vachris, 2004; Shen, Chung, Challis, & Cheung, 2007), most researchers agree (O'Toole & Absalom, 2003) that hybrid courses, when designed carefully, combine the best features of in-class teaching with the best features of e-learning to promote active student learning (Riffell & Sibley, 2005). Researchers found that technology can promote deeper exploration and integration of information and high-level thinking by allowing students to design, explore, experiment, and model complex and abstract phenomena (American Council on Education [ACE], 1999). According to Fjermestad, Hiltz, and Zhang (2005), students who connected abstract science to real-world problems through simulations, microcomputer-based laboratories, and videos obtained better results than students who experienced only traditional instructional methods. On the other hand, the traditional elements of hybrid learning preserve the non-fungible human touch of education. Furthermore, since hybrid learning treats students as individuals with different learning habits, learning styles and

preferences, it has the potential of considering some of the various learning needs (Irons, Keel, & Bielema, 2002; Beyth-Marom, Saporta, & Caspi, 2005).

Singh and Red (2001) define blended (or blending) learning as a learning programme where more than one delivery mode is being used with the objective of optimizing the learning outcome and the cost of programme delivery. According to Procter (2003), blended learning is the effective combination of different modes of delivery, models of teaching, and styles of learning. Valiathan (2002) describes blended learning as an optimal mixture of face-to-face classrooms, live e-learning, and self-paced learning. Other researchers define this learning method as the effective integration of various learning techniques, technologies, and delivery modalities to meet specific communication, knowledge sharing, and information needs (Finn & Bucci, 2004). According to the definitions, optimal hybrid/blended teaching-learning strategies have to take into account the principles of multi-sensory learning. Multi-sensory approaches promote variegation regarding the learning styles, teaching-learning methods, etc. Multi-sensory elements can facilitate careful design in hybrid courses and contribute to the effective combination, effective integration, and optimal mixture with respect to the blended learning.

2.4 Multimedia and multi-sensory learning at all levels

Daily life (in natural environments) exposes our brain to constant multi-sensory stimulation. As detailed above, recent research (Shams & Seitz, 2008) has demonstrated that the human brain learns and operates optimally in environments in which information is integrated across multiple sensory modalities. Since multi-sensory training protocols are closer to natural settings than the unisensory ones, they produce more effective learning. Young children, like some little scientists study their immediate surroundings in a very interactive way using all their senses. Interestingly, in line with current research results, a dominant current tendency in education is to simulate, even in academic environments (often making use of sophisticated technologies), children's way of learning: deep multi-sensory learning by doing (West, 1994).

Research in multimedia educational techniques goes hand in hand with the perceptual research of multi-sensory facilitation. Research in cognitive theory of multimedia learning (Harp & Mayer, 1998) adds further evidence to the conclusion that the mechanisms of multi-sensory facilitation can have important benefits in pedagogy (Shams & Seitz, 2008). Multimedia teaching-learning tools are changing the way students from all levels are taught in more and more educational institutes. New applications are daily integrated in the syllabus of almost

all educational fields. Through realistic animations, attractive musical sound, and vivid colours, abstract concepts are brought to life. In order to increase their impact, some of the software tools implement the so-called user-in-the-loop feature (Wong, Bigras, & Cervera, 2005). Special multimedia applications and computer games can increase students' motivation to learn and often lead to the better understanding of the studied topics (Philpot, Hall, Hubing, & Flori, 2005).

The experimental results of several researchers in virtual reality also indicate that converting data and abstract concepts into mutually reinforcing multi-sensory representations enhances students' understanding of scientific models (Loftin, Brooks, & Dede, 1998). This increasing realization of the cognitive importance of all of our senses is finding expression in several technologies. For example, with data sonification technologies, tables of numbers can be represented as sounds, revealing patterns in those data by changes in pitch and volume (the "music" produced would be an abstract but meaningful symphony of sound). In addition, there are companies which produce interfaces that convert digital data into different smells. A common characteristic of these applications is that they represent information that we usually do not perceive as having a sensory form (Staley, 2006).

More specifically, research has indicated that auditory aids can enhance the teaching process of the fractions (connecting fractions with musical notes) (Rawson, 1992). This approach to teaching fractions can be applied to other areas such as grammar. Since grammar is systematic in the same way that music is, teachers can work with students to understand "the melody line" of the sentences. Campbell (2000) discusses visual imaging in relation to spatial-temporal reasoning for mathematics and science concepts. His research on the "Mozart Effect" also serves as an example of the interconnectedness of the visual, auditory, and reasoning processes that occur within the human brain.

The methods we investigated during our first two studies explore in a harmonic way the visual, auditory, and kinaesthetic senses of the students. It helps them to imagine the studied abstract concepts and processes. In line with the above examples, the involved software tools use "structure sonification" or "recursive procedure/function sonification" to create "the melody line" of algorithms (Thompson, 2003). Students are also invited "to drum/type in the rhythm patterns of the loop skeletons of the algorithms" (using the keyboard) or "to play so-called recursive scenarios".

2.5 Multi-sensory learning through arts

Our third method takes additional multi-sensory elements into the programming education through arts (dance, music, rhythm, theatrical role-playing) too. Combining these art forms, teachers could create a multi-sensory learning

environment that involves almost all senses: visual, auditory, kinaesthetic, and tactile. It is not the artistic value of the methods that we want to emphasize. However, the presence of arts gives the class a touch of liveliness (beyond the cognitive benefits).

Dance means movements, patterns, music, and rhythms. Choreography is the art of making structures in which movement occurs. Dance is one of the most complex human activities involving the whole body and, what is more, the entire person (physical, cognitive, affective). As with dance, music is also characterized by repetitive rhythmic patterns. Additionally, during role-playing, actors follow scenarios that could also include patterns. Patterns and structures, as common elements in several art forms, represent the bridges between sciences and arts. For example, according to Hammel (2002), music is a logical structuring like a mathematical proof of itself. Stern, one of the initiators of the Math-Dance programme, stated that they translate pattern into choreography and pattern into math (Schaffer, Stern, & Kim, 2001).

2.5.1 Combining science education with arts

Since 1998, practising mathematicians, artists, musicians, and scientists have come together at the annual Bridges conferences to discuss connections existing among their fields of interest (Bridges Organization, 2004). In line with this initiative, in recent years, more and more papers have described works that combine science education with art.

2.5.1.1 *Science education on stage*

Downey uses his mathematical work as inspiration for the dances he choreographs and performs. His research includes algorithmic processes. In his opinion, since dancing means following a series of logical steps sequentially, Scottish country dances bear a striking resemblance to algorithms. Thinking about things moving in space, choreographers actually visualize algorithms (*The dance of mathematics*, 2006).

The Fibonacci and Phi and Une Journée Abstraite dance performances (initiated by Alban Elve'd Dance Company and worked out in collaboration with university scientists) create a fusion of mathematics, CS, graphical art, and dance. Fibonacci and Phi is played on the Fibonacci sequence and the Golden Ratio, Phi. Une Journée Abstraite introduces theoretical concepts of CS such as computability, language expressiveness, and Turing machines (Burg & Lüttringhaus, 2006). For the celebration of Einstein Year, in 2005, the Institute of Physics (UK) and Rambert Dance Company created Constant Speed, a performance inspired from the Einstein theories (Baldwin & Rivers, 2005). In 2006, Liz Lerman Dance Exchange Company premiered *Ferocious Beauty: Genome*,

an exploration of the complexity of genetics with regard to ancestry, aging, and diversity (Mtangi, 2006). Palindrome is another dance company that has adapted science to dance. Some pieces of their works are *DNA* (1981), *TRIO* (1989), and *Möbius Band* (1995) (Wechsler, 1997). Fishwick and his colleagues' (Fishwick, Diehl, Prophet, & Lowgren, 2005) work on aesthetic computing shows how algorithms and coding can be approached in terms of visual models with an artistry that provides alternative ways to understand computation.

In the case of the above-presented stage performances, professional artists provided the artistic elements. These productions demonstrate how science can be viewed as thematic element for dance performances. Further we present examples of how teachers who are not dance specialists or musicians implement the principles of multi-sensory learning through arts.

2.5.1.2 Arts in science classrooms

Combining mathematics and dance concepts, the Math-Dance programme makes it possible for audiences to experience a physical sensation of the abstract concepts of mathematics. Responding to requests coming from schools, they have extended their programme from the stage to classrooms. The Math-Dance project addresses teachers and students from primary grades to secondary and college level (Schaffer et al., 2001). The *Dancing the Words* research project aimed to develop children's language and conceptual understanding through dance lessons linked to their science curriculum (Moelwyn-Hughes, 2003).

In several New Mexico schools, teachers combine mathematics with teaching music and dance. Their experience shows that these two areas have much to offer to each other. Mathematics and music share a concern with numbers and patterns of change. In music and dance, these patterns are called rhythm, they said (Eisenhower SCIMAST, 1998). In *Teaching Science in the Primary Classroom*, the authors described how their students role-played solids, liquids, gases, aspects of sound, etc. (Ward, Hewlett, Roden, & Foreman, 2005). Chavey (1996) teaches algorithm analysis through song analysis.

According to Schaffer et al. (2001), the science-art combination is strongly recommended: (1) when a concept needs to be comprehended mentally, physically, and emotionally; (2) for the infusion of energy and excitement that can make students more receptive to learning; (3) in order to reach out to students that are mainly kinaesthetic learners. They stated that having a kinaesthetic experience of an abstract concept is very helpful in comprehending what that abstract is. They observed that students who generally are not very focused were highly engaged in lessons that integrated dance, and they enjoyed it. Since we have been applying multi-sensory methods in teaching-learning algorithms, we have smiling students at CS classes.

2.6 On the role of senses in education

Some conclusions of the above-presented research that had supported our expectation that the multi-sensory methods we designed have potential to enhance the teaching-learning process of computer algorithms are:

- The brain is organized to elaborate information, coming from the different sensory channels, cooperatively.
- Visual, auditory, and reasoning processes are interconnected.
- Multi-sensory structured methods have cognitive benefits.
- More senses mean more efficient teaching-learning process because:
 - more senses – more information,
 - different students – different dominant senses,
 - different students – different “intelligences”,
 - multiple senses – more pathways of locating the stored information,
 - multiple senses – distributed loading,
 - combined senses – more efficient learning process.

Consistent with these findings, Stevens and Goldberg (2001) stated that two of the core principles of brain-based learning are our brains’ desire for multi-sensory input; learning engages the whole body. Researchers emphasize that senses reach not only our feelings, emotions, and aesthetic sense but our intellect as well. In the opinion of medical neuroscientist Dave Warner, the traditional forms of information representation have been “perceptually deficient”, meaning that even multimedia digital content fails to consider “the extraordinary capacity of our brain to capture and process information from [all of] our senses” (Staley, 2006). According to Hung (2003), the recent findings in neuroscience have immediate implications for higher-level thinking skills (abstract problem solving, inference, deduction, and so on).

The didactical methods and software tools we are going to present explore these principles. The following expressions from the educational materials we created illustrate the key role of arts in the presented methods: dancing algorithms, the melody line of the recursion, playing recursive scenarios, rhythm of the algorithms, drumming-in algorithm skeletons, piano accompanying the algorithms, etc. We believe that our effort to enrich blended/hybrid learning with multi-sensory elements implemented through arts has resulted in a kind of cocktail of learning. Like a cocktail drink, the resulted teaching-learning strategies are characterized by multiple variegations and are both instructive (nutritious) and fun (exciting).

3 SEEING, HEARING, AND TOUCHING COMPUTER ALGORITHMS (STUDY 1)

Since computer algorithms are abstract processes, instructors use a variety of instruments to make them perceptible to learners. The most common educational tools use visual representations. A next, quite challenging step could be making the algorithms perceptible for the auditory sensory. In addition, connecting tactile senses to the learning environment can be even more challenging. Our first study focused on this topic (Kátai, Juhász, & Adorjáni, 2008). In this chapter, we present (1) the multisensory method and tool we developed to support the teaching-learning process of elementary algorithms and (2) the investigation we performed.

3.1 Anatomy of simple algorithms

The majority of algorithms have a “loop skeleton”, its structure of loops. The instructions that represent the nucleus of the loops can be seen as the “meat parts” of the algorithm. In what follows, we recommend a two-step method for teaching and learning simple algorithms:

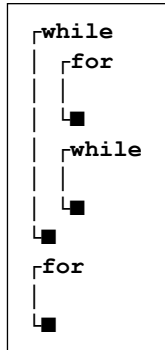
1. By analysing the task, we establish the loop skeleton of the algorithm that solves the problem.
2. We fill up the loop skeleton with the adequate instructions.

Since the second step presumes the first, the teacher – when the problem solving takes place under his/her supervision – should not allow the implementation until the students comprehend clearly the loop skeleton of the algorithm. In the following, we will illustrate the above method through a sample problem.

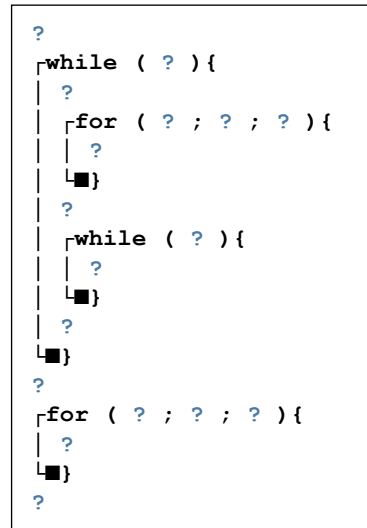
Problem: Write a C/C++ program that reads natural numbers from the keyboard until a zero number appears and then verifies whether the sum of the products of the digits of the prime numbers is prime or not (see *Figure 3.1*).

We can help students to make the second step with the following Pólya-type (1945) question sequence:

- What are the input/output data of the problem?
- What kind of variables (data structures) should we use to store the input data?
- Where do the input data reading and the output data writing have to take place in the framework of the algorithm?
- What sub-problems do the “inner loops” have to solve?
- What auxiliary variables are required to solve the sub-problems?
- What (the nucleus of the loop) and while (the condition of the loop) do the particular loop statements have to repeat?



(a)



(b)

```

s = 0; cin >> nr;
while ( nr != 0 ){
    prime = 1;
    for ( i = 2 ; i <= sqrt(nr) ; ++i ){
        if ( nr % i == 0 ) { prime = 0; break; }
        █
    }
    if ( nr > 1 && prime == 1 ){
        p = 1;
        while ( nr != 0 ){
            p *= nr % 10; nr /= 10;
            █
        }
        s += p;
    }
    cin >> nr;
    █
}
prime = 1;
for ( i = 2 ; i <= sqrt(s) ; ++i ){
    if ( s % i == 0 ) { prime = 0; break; }
    █
}
if ( s > 1 && prime == 1 ){ cout << "PRIME"; }
else { cout << "NOT PRIME"; }

```

(c)

Figure 3.1. The loop skeleton of the algorithm and the equivalent C/C++ program: (a) step 1; (b) intermediate step; (c) step 3

- Where do the initializations belonging to the certain sub-problems have to take place (before which loop statements)?

In the followings, we will focus our attention on the first step of the presented method, namely the way we can help students develop the skill of recognizing the loop skeleton of the algorithm. Since this phase of the method implies a developed abstraction skill, we have proposed to create a software tool that makes multiple-sense involvement possible.

3.2 Software tool

The application we have developed has four main modules: `code_creator`, `code_beautifier`, `code_buherator`, and `run_code`.

The `code_creator` module (see *Figure 3.2*) makes it possible to create program skeletons with different loop structures in an automatic way. The attached figure shows the user interface of this module. It runs in two modes:

- Giving the parameters of the loop skeleton: We introduce – in the columns labelled with levels I, II, and III – how many loops we want on the first, second, and third level and which is subordinate to which. Additionally, we need to give the number of iterations of each loop. In the sample from *Figure 3.2*, the `code_creator` module will generate a code that has two first-level loops (with 2 and 5 iterations), and the first of them has two subordinate loops on the second level (with 3 and 4 iterations).
- Drumming the loop skeleton in: This mode is supervised by the Drumming Area of the dialogue box, making it possible to type in the loop skeleton of the program, as if we have drummed in its rhythm pattern. For the first-, second-, and third-level loops' drumming in, we implicitly use the keys a, f, and j. The above-presented loop skeleton has the following drum rhythm (The ‘_’ characters mark the space keys which must be introduced between two loops that follow in succession on the same level):

```
afff_ffff afff_ffff aaaaa
```

Pushing the **Apply** button, the C/C++ program will be automatically generated, which we can see on the right side of the display. Kinaesthesia is involved especially at this stage of the learning process.

By the `code_beautifier` module, every C/C++ source file can automatically be reorganized (“beautified”) in such a way that its loop skeleton should easily be noticed. This operation is given an important role because of eyesight involvement (see *Figure 3.4*).

The `code_buherator` module – by rewriting the source file – plants sound and delay procedures in the nuclei of each loop instruction.

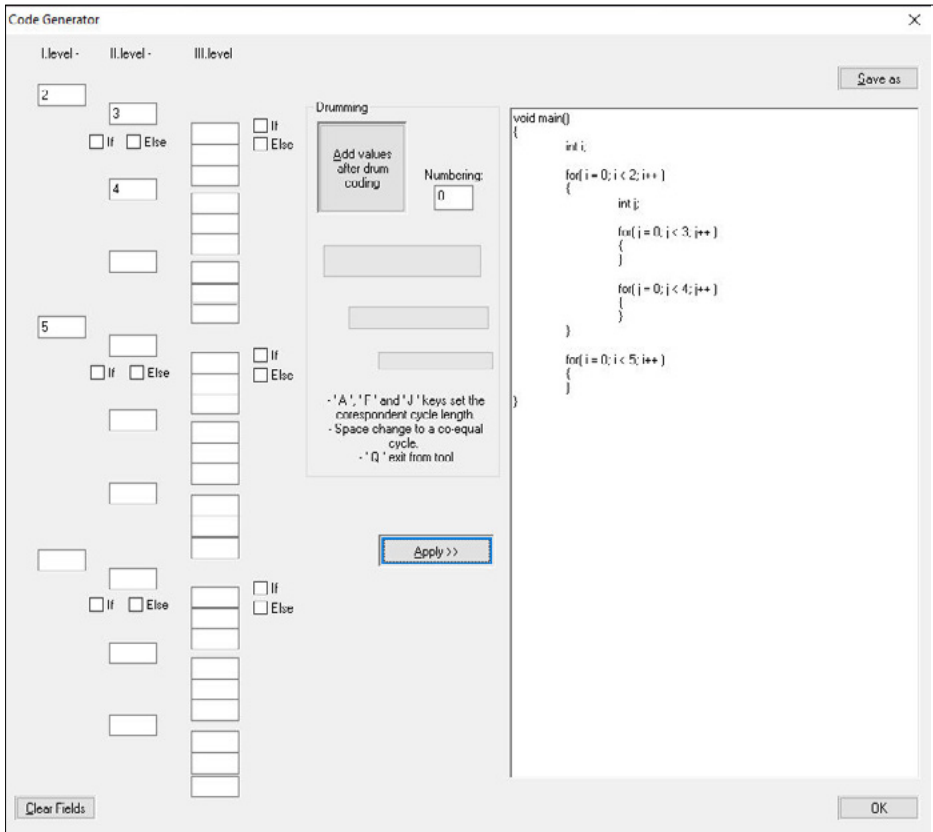


Figure 3.2. *The dialog box of the code_creator module*

As a result, a piano sound will be heard every time when the nucleus of a loop is traversed. The outer loops will be audible in a lower pitch and the inner ones in a higher pitch. Additionally, by applying different length delays in the case of the loops situated on different levels, the result will be that the outer loops will have smaller frequency sound sequences than the inner ones. For instance, the above-presented loop skeleton will be audible as it follows (do, fa, and si sounds have been built into the I, II, and III. level loops; the ‘_’ characters represent the lengths of pauses):

do fa_fa_fa_fa_fa_fa do fa_fa_fa_fa_fa_fa do__do__do__do__do

When the algorithm has loops in both branches of a selection, we have “parallel loops” at the same level. So, depending on the condition, we will hear them by turns.

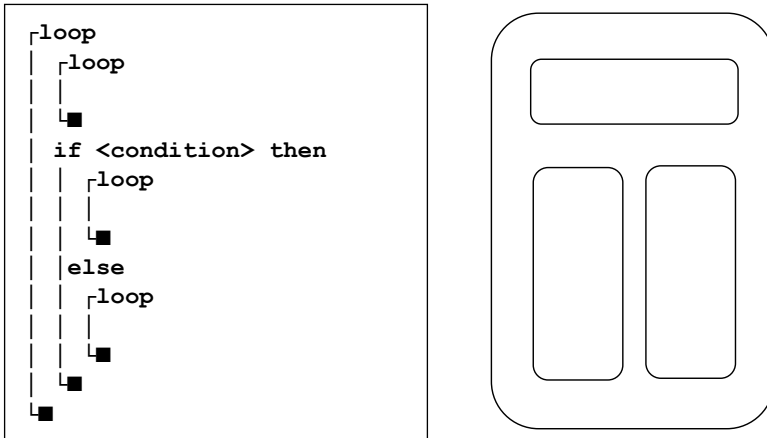


Figure 3.3. A loop skeleton example and its representation

How will the listener discern the loops, and how will he/she be able to distinguish them? The solution we have chosen is the following: in parallel loops, we implemented the same sounds but with different musical instruments. For example (see *Figure 3.3*):

A possible sound sequence of the above algorithm is (fa* is a violin fa):

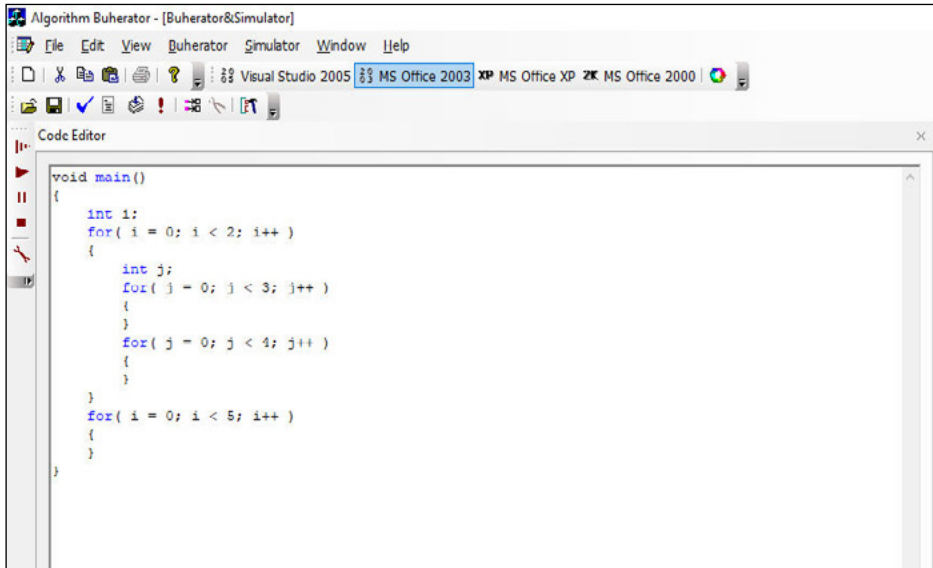
do fa_fa_fa_fa_fa_fa_fa_fa do fa_fa_fa_fa*_fa*_fa*_fa*_fa*

The nucleus of the outer loop is repeated twice, the nucleus of the first inner loop three times (during its both executions), and the nuclei of the parallel inner loops five and four times respectively. Each of the parallel loops is executed ones.

In the dialogue box of the `run_code` module (see *Figure 3.4*), the “beautified C/C++ code” of the analysed algorithm appears. Pushing the **Run** button starts the slow-motion running of the program. While the students “are listening to the loop skeleton of the algorithm” represented by its sound sequence, they can keep their eyes on the program’s running (as we can see, the instruction which is being executed is highlighted).

3.3 Suggested syllabus

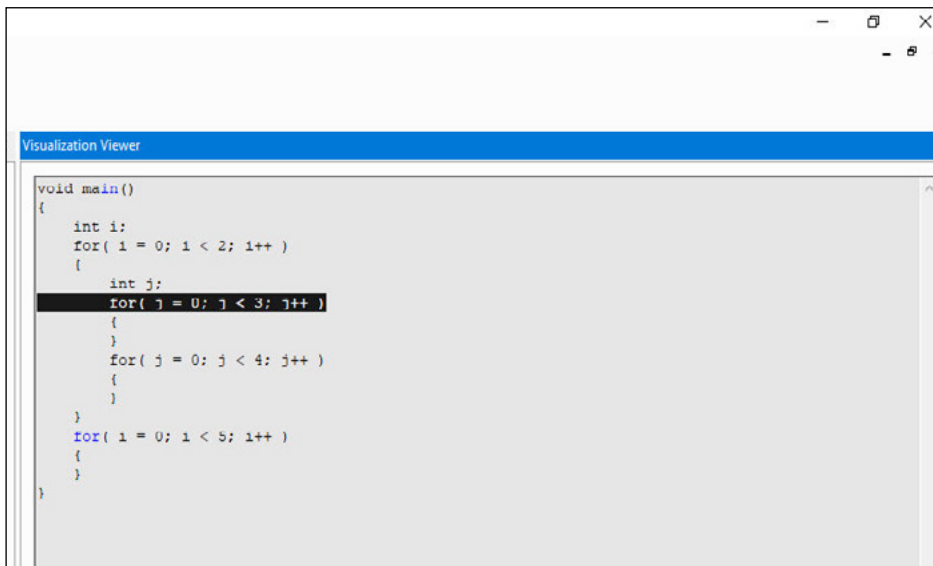
We suggest the following syllabus (students are not only observers of a simulation, they are actively involved in the teaching-learning process; bidirectional student–computer communication):



The screenshot shows the 'Algorithm Buherator - [Buherator&Simulator]' application window. The title bar includes 'File', 'Edit', 'View', 'Buherator', 'Simulator', 'Window', and 'Help'. The menu bar contains 'File', 'Edit', 'View', 'Buherator', 'Simulator', 'Window', and 'Help'. The toolbar includes icons for file operations and execution. The 'Code Editor' window displays the following C++ code:

```
void main()
{
    int i;
    for( i = 0; i < 2; i++ )
    {
        int j;
        for( j = 0; j < 3; j++ )
        {
            for( j = 0; j < 4; j++ )
            {
            }
        }
        for( i = 0; i < 5; i++ )
        {
        }
    }
}
```

(a)



The screenshot shows the 'Visualization Viewer' application window. The title bar includes standard window controls. The 'Visualization Viewer' window displays the same C++ code as in (a), with the innermost loop highlighted in black:

```
void main()
{
    int i;
    for( i = 0; i < 2; i++ )
    {
        int j;
        for( j = 0; j < 3; j++ )
        {
            for( j = 0; j < 4; j++ )
            {
            }
        }
        for( i = 0; i < 5; i++ )
        {
        }
    }
}
```

(b)

Figure 3.4. The user interface of the run_code module

- Students “are listening to” several loop skeletons while they keep their eyes on the slow-motion running of the program generated by the `code_creator` module. It would be useful to analyse the following loop skeletons (see *Figure 3.5*).
- Students are asked to recognize some unknown loop skeletons only by hearing their “piano accompaniment”.
- Students are following, with their eyes and ears, the running of some well-known basic algorithms in the dialogue box of the `run_code` module. For example, how does a searching algorithm sound? And what about a sorting algorithm? If, for instance, students remember the sound pattern of sorting algorithms, they would not try to create a single loop.
- Students are listening to the “piano accompaniment” of the algorithm for different inputs.
- Students are asked to “drum in” divers loop skeletons. This is a very important stage of the syllabus. As the pupils in New Mexico schools feel the fractions in their bones due to the music and dance, our students should get to the point where the rhythm of algorithms rings in them. They reach this stage as a result of using their fingers in the manner of a pianist to “type” again and again the loop skeleton of the different algorithms. This phase of the method can be applied even if no computers are accessible. Firstly, the teacher and then the students can “drum” the rhythm of the loop skeletons, using their hands and legs.
- Algorithms may contain typical errors. Students are asked to compare the wrong sound sequences with the “piano accompaniment” of the correct algorithms.
- A certain loop skeleton is selected, and problems with the respective skeleton are analysed. Students are also asked to suggest adequate problems (preferably real-word problems).

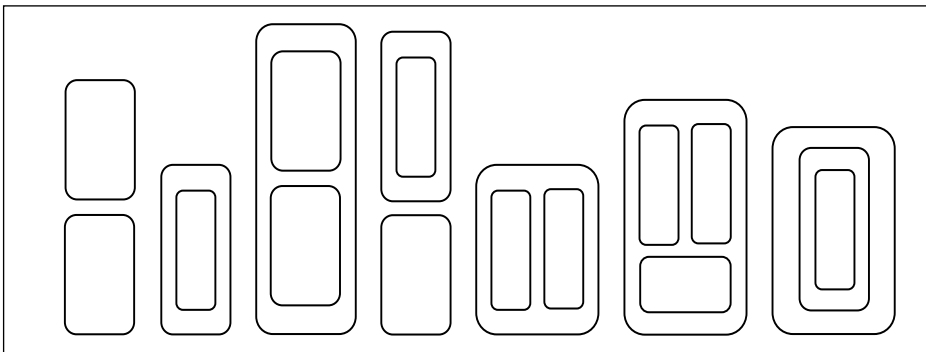


Figure 3.5. *Loop skeletons*

3.4 The experiment

In order to prove empirically the efficiency of the above-presented didactical method, we performed the following experiment. Two 9th grade classes (IX. G and IX. H) were involved in the experiment (with 24 students each) from Bolyai Farkas High School (Târgu-Mureş, Romania). Both classes started to learn C/C++ programming language at the beginning of the 2005/2006 school year. The experiment was ran at the end of the first term. The two classes were taught by different teachers but according to the same syllabus.

Table 3.1. *The averages of the groups before and after the experiment*

	First-term averages	Means of the test points
G – control group	8.00	4.78
G – experimental group	8.00	6.24
H – control group	7.83	5.26
H – experimental group	7.83	6.30

Source: Kátai, Juhász, Adorjáni, 2008

As a preparation of the experiment, we organized a pre-test in order to compare the classes. According to the pre-test results, the average of class H was better than the average of class G by nearly one mark on the 1–10 marking scale, which is used in Romania. Both classes were divided into two “equivalent” groups (an experimental group and a control one) from the beginning of the school year (we verified the equivalence of these groups in both classes). This comparison was made on the basis of the students’ performance during the whole term. As we can see below, the groups could be considered “equivalent” (see *Table 3.1*). We decided to identify the students with their position in the sorted list of their group.

During the two-week experiment, in the experimental groups of each class, the simple algorithms subject was taught according to the above-presented two-step method and syllabus. In the control groups, evidently, the students were taught according to the classic methods, without making any effort to involve the senses in the teaching-learning process.

As our main goal was to check whether the two-week application of the method could lead to the deeper understanding of the principles behind the algorithms and a more flexible knowledge, we deliberately made the test quite difficult and unusual. In the followings, we have listed the kind of problems the students of both classes had to solve:

- Students received different loop skeletons, and they had to think of such problems whose algorithms have the same loop skeletons (2 points).

- We muddled the lines of a given problem’s algorithm, and the students had to establish the correct line order (3 points).
- Students had to determine the loop skeleton of a difficult problem’s algorithm (without explicitly writing the algorithm) (2 points).
- Students had to check the loop skeleton of a fairly difficult problem’s algorithm and then write explicitly the C/C++ code that resolves the problem (2 points).
- The 10th point was received for filling in the test.

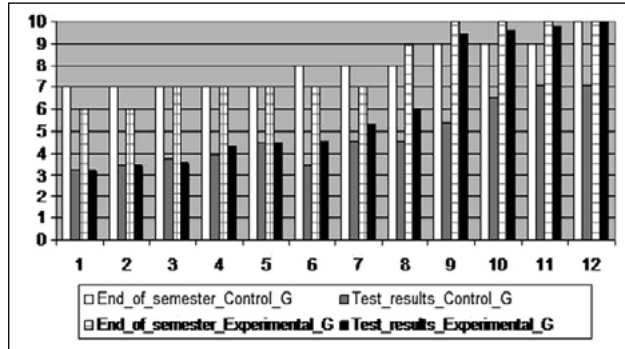
3.4.1 Results and discussion

The diagrams below show the results of the experiment (see *figures 3.6–7*). Students (identified by their position in the sorted list of the group they belong to) are represented on the horizontal axis, and their results on the vertical axis. The white and grid columns represent the first semester average (*End_of_semester_mark*) of the experimental and the control group, respectively, and the grey and black columns represent their test results after the experiment (*Number_of_test_points*).

The means of the points the students belonging to the experimental (12 + 12 members) and control (12 + 12 members) groups received for the test are 6.27 and 5.02 respectively. Comparing these values with the independent samples’ t-test, we found that the difference between them is significant ($p = 0.038 < 0.05$) (favouring the experimental group).

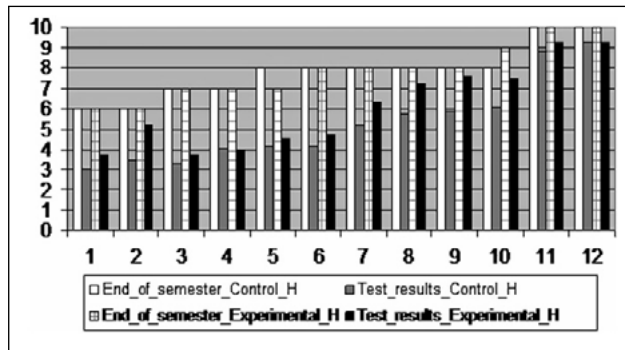
As we can discern from the figures, the points the students received for the test are consistently below the marks they had on the basis of their whole semester performance. This was a direct consequence of the nature of the test. Although the lower results as a phenomenon can be observed in the case of all students, the size of these differences (*End_of_semester_mark* – *Number_of_test_points*) varies from student to student. Since we expected that the new method was going to enhance the students’ abilities to apply their knowledge even in “unfamiliar circumstances”, we also decided to analyse these differences. The means of the differences for each group in part are: 2.56 (control group in class H), 1.53 (experimental group in class H), 3.21 (control group in class G), 1.75 (experimental group in class G). Comparing these values with the independent sample t-test (in the case of each experimental–control group pair), we found that the differences are significant (favouring the experimental groups): for class H, $p = 0.016$; for class G, $p = 0.002$. Then again, when comparing the performances of the two control groups or the two experimental groups, we did not receive significant differences. (The better test results in the case of the experimental groups cannot be explained by the contingent differences between the teachers.)

In conclusion, it can be stated that the results of this didactical experiment support our expectation that the multi-sensory method presented above improves students’ skill to analyse and design algorithms.



Source: Kátai, Juhász, Adorjáni, 2008

Figure 3.6. *The results of class G
(12 students each in the experimental group and the control group)*



Source: Kátai, Juhász, Adorjáni, 2008

Figure 3.7. *The results of class H
(12 students each in the experimental group and the control group)*

Additionally, teachers involved in the experiment state that the division of the computer programming process in two well-delimited stages (the right loop skeleton identification and its filling with the proper instructions) taught the students how useful is to design even the simplest algorithms. In other words, the presented method can be seen as a promoter of the widely applied five steps problem-solving process (1. abstraction, 2. decomposition, 3. transformation into sub-solutions, 4. recomposition into a working program, 5. evaluation). This approach has led to less mistaken algorithms. The number of such mistakes when the student knew which instructions had to be used but did not put them on the right place within the algorithm was also reduced.

The already mentioned New Mexico schools, where the “rhythms of mathematics” method is applied, usually organize an end-of-year ceremony for parents, where the pupils demonstrate their skills. Using this idea, we can ask the “musician students” (the ones who play musical instruments) to prepare a “piece” for the next class. The colleagues will have to identify the algorithms heard. It is not hard to imagine the positive emotional effect on the students when they hear their own colleagues playing the guitar or the flute in CS classes.

4 PLAYING RECURSIVE SCENARIOS (STUDY 2)

Our second research focused on the teaching-learning process of recursive algorithms (Kátai, 2011). We have proposed to extend the above described method with didactical role-playing. To help students imagine how recursion works, they were invited to play the running process of certain recursive functions/procedures. In the following, we will detail this method and the investigation we performed.

4.1 Teaching/learning recursion

Recursion in CS is a way of thinking about and solving problems. It is, in fact, one of the central ideas of CS. Solving a problem using recursion means the solution depends on solutions to smaller instances of the same problem.

Virtually, all programming languages in use today allow the direct specification of recursive functions and procedures. When such a function is called, the computer or the language implementation keeps track of the various instances of the function.

Creating a recursive procedure/function essentially requires defining a “base case” and then defining rules to break down more complex cases into the base case. Key to a recursive procedure/function is that with each recursive call the problem domain must be reduced in such a way that eventually the base case is arrived at. Without such a termination condition, a recursion would go on forever.

We propose two different strategies to teach the concept of recursion: one for the procedures and the other for the functions. (Usually procedures execute tasks, and functions calculate and return values.)

4.1.1 How to teach recursive procedure design?

The termination condition is usually implemented by an if instruction (we will identify this if instruction as the “key if”). If we imagine the recursive procedure as the one that is built around this “key if”, we get the following “recursive-procedure-skeleton” (see *Figure 4.1*). Any instruction can be placed only in the specified zones.

- zone a: Instructions placed before the “key if”;
- zone A: Instructions placed after the “key if”;
- zone b: Instructions placed on the recursive branch of the “key if”, before the recursive call;
- zone B: Instructions placed on the recursive branch of the “key if”, after the recursive call.

- zone X: Instructions placed on the non-recursive branch of the “key if” (the “base case”)

```

void RP( <formal parameters of the general problem> ){
    [[ zone a ]]
    if ( <condition> ){
        [[ zone b ]]
        RP ( <effective parameters of the sub-problem> );
        [[ zone B ]]
    }
    else{
        [[ zone X ]]
    }
    [[ zone A ]]
}

```

Figure 4.1. *Recursive procedure skeleton*

Assume that the above “recursive-procedure-skeleton” is called recursively n times (the first $n-1$ calls are recursive, and the last one is non-recursive). The above-defined zones (the instructions placed in these zones) are executed as many times and in the order as it follows. (The indexes represent the instances (the processing levels) of the recursive procedure; the letter c represents the evaluation of the condition of the “key if” instruction; T and F mean True and False.)

$$a_1 c^T b_1 a_2 c^T b_2 \dots a_{n-1} c^T b_{n-1} a_n c^F X A_n B_{n-1} A_{n-1} \dots B_2 A_2 B_1 A_1$$

Remark: At first, the “a-zone instructions” (a_1) and then the “b-zone instructions” (b_1) of the first instance of the procedure are executed. After this, due to the recursive call, the first instance of the procedure is suspended, and RP is run for the second time. This second instance also starts by executing the “a-zone” (a_2) and then the “b-zone” (b_2) instructions of the procedure and so on. Firstly, the n^{th} instance is ended ($a_n X A_n$), and after this the $(n-1)^{\text{th}}$ instance is continued by executing the “B-zone instructions” (B_{n-1}) and then the “A zone instructions” (A_{n-1}) of the procedure and so on.

According to our experience, the above-presented approach of the recursive procedures helps in a deeper understanding of the concept of recursion. Students realize that the number and the order in which the instructions of the recursive procedures are executed directly depend on the zone they are placed:

- zone a: is executed n times in the order of the recursive calls,
- zone A: is executed n times in the reverse order of the recursive calls,
- zone b: is executed $(n-1)$ times in the order of the recursive calls,

- zone B: is executed $(n-1)$ times in the reverse order of the recursive calls,
- zone X: is executed only once in the last instance of the procedure.

Students should fill in this skeleton after they have established the order/number according to which the certain instructions have to be executed/repeated (according to the task the recursive procedure has to perform).

4.1.2 How to teach recursive function design?

The recursive function design process should start with the following question: How can the problem be reduced to similar, simpler sub-problem(s), then later again reduce these ones to similar, even simpler sub-problem(s) until we get trivial sub-problems (“base case”)? Answering this question, students will hopefully be able to determine the general (parametric) form of the sub-problems. The parameters of this general form of the problem will constitute the formal parameters of the function. If the problem which the function has to solve is trivial, then the result is calculated (without recursive call) and returned. Otherwise, the problem is handled recursively. Since the recursive function has to handle both trivial and non-trivial sub-problems, two “scenarios” have to be implemented: a “trivial scenario” (according to the effective parameters of the current call, the problem that has to be solved is trivial) and a “recursive scenario” (according to the effective parameters of the current call, the problem that has to be solved is non-trivial).

As a rule, the recursive scenario has two parts: a “recursive part” and an “own part”. According to this partition, the task which the current instance of the function has to solve is also divided into two parts. The recursive part bears the brunt of the task by transferring it to the next instance of the function. In other words, the current instance of the function, by a recursive call, asks for the result of the lion’s share of task “on a plate”. This transferred part obviously has to be a similar and simpler sub-problem (why? (1) to be manageable by the next instance of the same function and (2) to “converge” to the base case). Once the next instance of the function has “delivered” this result, the current instance of the function fulfils its own part by building up the solution of its entire task.

The above-presented approach of the recursive functions inspires the following “recursive-function-skeleton” (see *Figure 4.2*).

This skeleton can be filled in as follows:

1. Defining the formal parameters of the function and determining the parameters of the general problem (sub-problems) go hand in hand.
2. Students establish the condition of triviality.
3. Students implement the “trivial scenario”.
4. How can the general form of the problem be reduced to a similar, simpler sub-problem? Students determine the effective parameters of the recursive call on the basis of the formal parameters (the “recursive part” of the “recursive scenario”).

```

<type> RF( <formal parameters of the general problem> ){
  <type> plate;
  if ( <condition of the triviality> ){
    [[ the "trivial scenario" ]]
  }
  else{
    plate = RF ( <effective parameters of the sub-problem> );
    [[ the "own part" of the "recursive scenario" ]]
  }
}

```

Figure 4.2. *Recursive procedure skeleton*

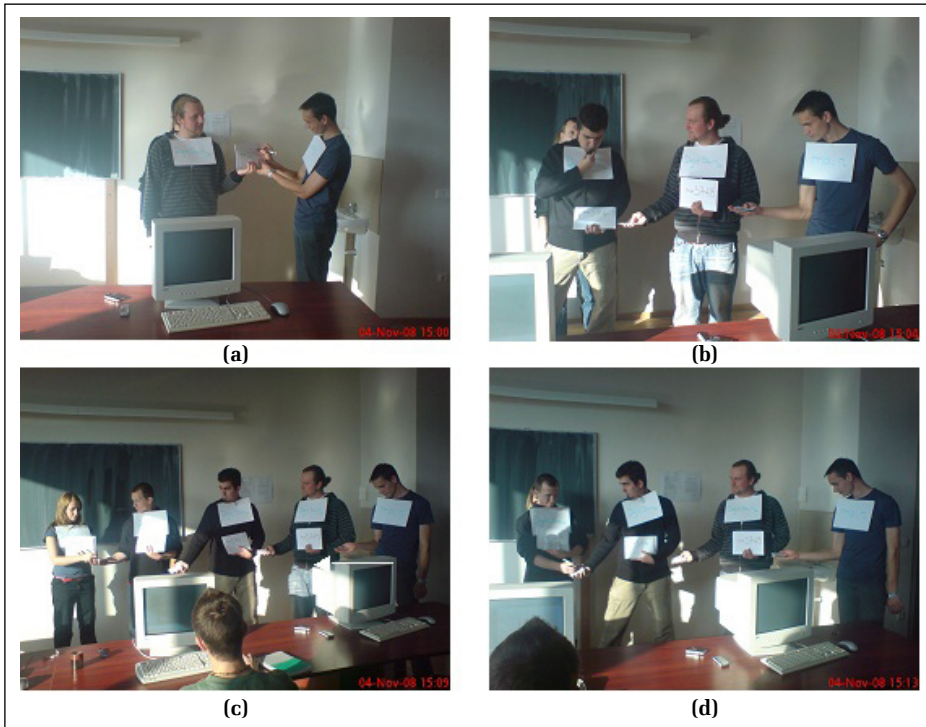
5. Students implement the way the solution of the problem (described by the formal parameters) can be determined from the value of the “plate variable” (the “own part” of the “recursive scenario”).

Remark: In the case of the skeleton presented in *Figure 4.2*, zones a, b, and A are empty. Additionally, the trivial scenario and the “own part” of the “recursive scenario” are implemented in zones X and B respectively. Sometimes it is more efficient to implement the “own part” in zone b.

Although the above described “recursive-procedure-skeleton” and “recursive-function-skeleton” can be applied only in particular cases, they are useful in helping students to understand the mechanism of the recursion. (In the general case, the recursive procedures/functions may contain several recursive calls, and they may have almost any kind of structure.)

4.2 How can the presented methods be improved by kinaesthesia?

Students are invited “to play the running process” of certain recursive functions/procedures. Let us suppose that the recursive function the teacher has chosen as an example returns the sum of the digits of a natural number (see *Figure 4.8a*), and he/she wants to demonstrate its functioning for a four-digit number. For this “scenario”, we need five actors. One of the students plays the role of function main. Other four students personify the four instances of the recursive function (“recursive actors”). The formal parameter (n) and the local variable (plate) of each instance of function DigitSum are represented by two pieces of paper in the hands of the corresponding students (n-sheet, plate sheet).



Source: author's own pictures

Figure 4.3. *Staging a recursive scene: (a) First-instance student receives his/her task from student main; (b) Second-instance student analyses his/her task (recursive or trivial scenario); (c) Last-instance student simply returns the result; (d) Third-instance student computes and returns the result*

Once a “recursive actor” “was called” (the “called-actor” steps out from behind the “caller-actor”), his/her task (the number s /he has to process) is written on his/her n -sheet by his/her “caller function classmate”. The “recursive actor” in focus analyses the received task and chooses if s /he will follow the recursive or trivial scenario (the first three instances “choose” the recursive scenario and the fourth the trivial one). According to the recursive scenario, the student in focus “activates” (“calls”) his/her classmate, who plays the next instance of the function, writes value $n/10$ on the n -sheet thereof, and begins to wait with the plate sheet in his/her right hand. During the waiting periods, the “suspended actors” “are petrified”. According to the “trivial scenario”, the student who plays the last instance of the function simply copies “his/her n -value” to the plate sheet of his/her “caller function classmate”. After the focus has returned

to a certain “recursive actor”, this student adds the last digit of his/her n -value to the value s /he received through his/her plate sheet and writes the result on the plate sheet of the student who plays the role of the previous instance. Once a “recursive actor” has finished his/her role, s /he returns behind his/her “caller function classmate”. The “piece” ends when the “main-function-student” receives the result (the digit sum of the four-digit number) from the “first-instance student”. *Figure 4.3* shows snapshots from the “staging” of the presented scene.

By inviting students to play recursive scenarios, the teacher helps them to imagine how recursion works. Whereas the spectator students get an overview of the whole process, the actor students “memorize in their muscles” (kinaesthetic memory) specific moves associated with elementary operations related to the recursive scenarios. (Kinaesthetic perception is based on information from muscles, tendons, and joints, and therefore physical movement triggers it. When spectator students watch their actor classmates’ movements, the process of kinaesthetic memory works reversely, from (visual) image to muscle memory.) (Moen, 2008).

4.3 How can the presented methods be improved by audio-visual elements?

The software initially attached only sounds to the recursive subprograms, the same sounds for all instructions from a certain level (i) of the recursive process. Between successive processing levels (instance of the procedure), a constant shift was applied (on the score of pitch). The recursive calls and the returns were accompanied by special sound effects (\downarrow, \uparrow). At a certain processing level, the instructions from different zones (a, A, b, B, X) sounded the same although coming from different musical instruments. In *Figure 4.4*, we can see a sample recursive procedure that contains empty instructions in its zones. Running this program, the following sound sequence was heard:

$$\begin{aligned} &\downarrow a_0 a_0 a_0 \ b_0 b_0 b_0 \downarrow a_1 a_1 a_1 \ b_1 b_1 b_1 \downarrow a_2 a_2 a_2 \ b_2 b_2 b_2 \downarrow \\ & a_3 a_3 a_3 \ X_3 X_3 X_3 X_3 X_3 \ A_3 A_3 A_3 \uparrow \\ & B_2 B_2 B_2 B_2 \ A_2 A_2 A_2 \uparrow B_1 B_1 B_1 B_1 \ A_1 A_1 A_1 \uparrow B_0 B_0 B_0 B_0 \ A_0 A_0 A_0 \uparrow \end{aligned}$$

Although this kind of sonification of the recursive procedures helped students to detect the order/number in which the instructions are executed/repeated, they found this “melody of the recursion” primitive and unenjoyable (Kátai, Juhász, & Adorjáni, 2008).

```

void R ( int i, int n ){
    ; ; ; //[[ zone a ]]
    if ( i < n ){
        ; ; ; ; //[[ zone b ]]
        R ( i + 1, n );
        ; ; ; ; //[[ zone B ]]
    }
    else{
        ; ; ; ; ; //[[ zone X ]]
    }
    ; ; ; //[[ zone A ]]
}
void main{
    R(0, 3);
}

```

Figure 4.4. Sample recursive procedure

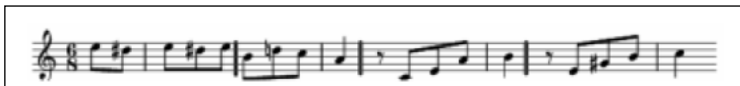
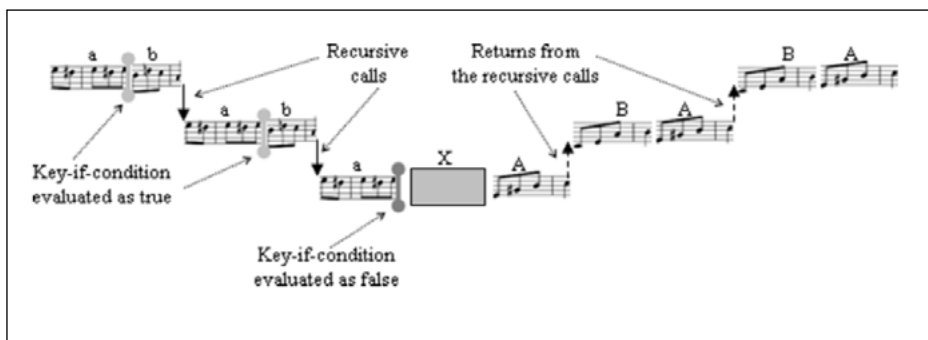


Figure 4.5. Fragment from Für Elise by Ludwig von Beethoven



Source: Kátai, 2011

Figure 4.6. The staircase illustration of a three-level recursive call

It can be observed that the number and the type of the instruction placed in the different zones of the skeletons have no importance with respect to understanding the mechanism of the recursion. Consequently, we decided to use segments from the classical piano masterpiece *Für Elise* by Ludwig von Beethoven as accompanying music for the execution of the instructions from zones a, b, A, B. We divided the below presented music fragment into four segments according to the number of the zones on the recursive branch of the “recursive procedure skeleton” (see *Figure 4.5*). Between successive processing levels, a constant shift is applied (on the score of pitch).

For zone X, we chose a different music fragment. The evaluations of the “key if” instruction (depending on the logical value of the condition), the recursive calls, and the returns from the recursive calls are also accompanied by the sound effects. We will refer to these phases of the run of the recursive procedure as key momentums. According to the so-called “staircase illustration method”, the mechanism of the recursion can be represented as shown in *Figure 4.6*. The successive calls of the recursive procedure/function are represented on “successive floors”. A given music segment’s sound at different pitch ranges on different floors (the relative shift between successive floors is one note):

Highlighting the staircase character of the recursive actors’ step pattern, the teacher makes connection between the audio-visual elements of the “recursive movies” and the kinaesthetic elements of the “recursive pieces” (see *Figure 4.3.a–d*).

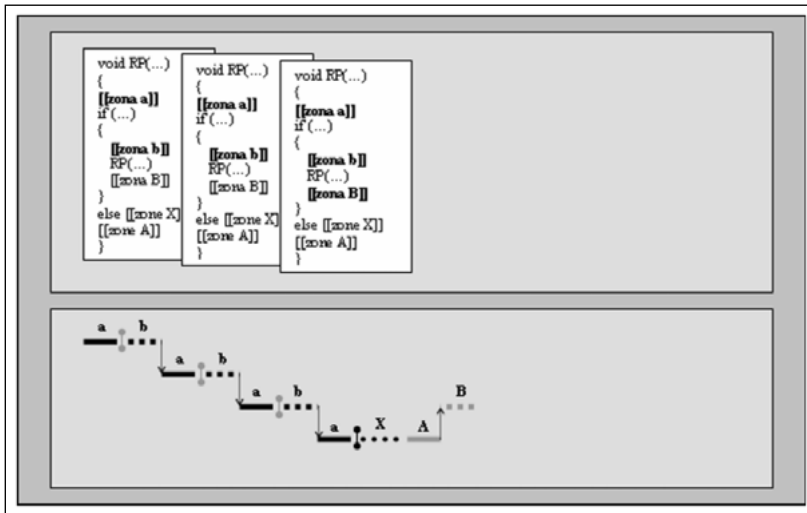
4.4 Software tool and suggested syllabus

The software we developed for the teaching-learning process of the recursion has three main modules: `code_beautifier`, `code_buherator`, `run_code`.

By the `code_beautifier` module, every C source file can automatically be reorganized (“beautified”) in such a way that the zones (and the instructions placed in these zones) of the recursive procedure/function should be easily noticed. This operation gets an important role because of the eyesight involvement.

The `code_buherator` module – rewriting the beautified source file – includes `fwrite` instructions in the code of the recursive procedure/function: at the beginning of the zones, before and after the recursive call, etc. Running this modified code, an output text-file is generated (by means of the `fwrite` instructions) which contains the necessary information for the simulation process of the recursive procedure/function.

In the dialogue box of the `run_code` module, the “beautified C code” and the attached “staircase” appear side by side (see *Figure 4.7*). During the simulation process, the following multi-sensory phenomenon comes to fruition: while the students are listening “the accompanying music” of the slow-motion running of the recursive procedure/function, they keep one eye on the program’s running



Source: Kátai, 2011

Figure 4.7. *The dialogue box of the run_code module*

<pre> int DigitSum(int n){ if (n < 10){ return n; } else{ int plate = DigitSum(n/10); return plate + n%10; } } </pre>	
--	--

(a)

(b)

Source: Kátai, 2011

Figure 4.8. (a) *Recursive function that returns the digit sum of a natural number, (b) representation of the accompanying music of the recursive function and its interpretation*

(the instruction which is being executed is highlighted) and the other one on the aligned traversing process of the attached “staircase”.

Teachers should include the following items in the syllabus of the teaching recursive procedures/functions:

- The students “are viewing the movie” of several recursive procedures/functions. The teacher is the narrator, and s/he should give interpretations depending on the type of the subprogram (procedure or function).
- Consider the following example: the task is to write a recursive function that returns the digit sum of a natural number. *Figure 4.8.a,b* shows the C code of the function, the codified accompanying symphony of the sounds, and a possible interpretation for this.
- Students are asked to draw the “staircase representation” of the execution of some unknown recursive procedures/functions only by hearing their “accompanying music”.

4.5 The experiment

In order to prove empirically the efficiency of the above-presented didactical method, we performed the following experiment. Forty-three first-year undergraduate computer science/electrical engineering/mechatronic students were involved in the experiment from Sapiientia Hungarian University of Transylvania (Târgu-Mureş, Romania). All students started to learn C programming language at the beginning of the 2008/2009 school year. The experiment took place in the 7th and 8th weeks of the first semester, after students having been initiated in working with function and learned to use loop instruction in programming repetitive structures. Our goal was to find out if they could assimilate (at this early stage of their computer science course) the mechanism of the recursion as an alternative method to implement repetitive structures.

As a preparation of the experiment, we organized a pre-test in order to form two “statistically equivalent” groups (experimental/control). In the pre-test, we tested students’ knowledge in applying loop instructions. The means of the pre-test points of the students assigned to the experimental and control groups are 7.01 and 7.00 respectively (according to the 1–10 marking scale in use in Romania).

During the two-week experiment for the 20 members of the experimental group, recursion was taught according to the new multi-sensory method, following the above syllabus. (In the control group, evidently, the 23 students were taught according to “classic methods”, without making any effort to involve the senses in the teaching-learning process.)

- Teacher presents the “recursive procedure skeleton”.
- Students “watch the movie” of several recursive procedures/functions.
- Students are asked to draw the “staircase representation” of the execution of some unknown recursive procedures only by hearing their “accompanying music”.
- Students practice designing and implementing recursive procedures.
- Teacher presents the “recursive function skeleton”.

- Students are invited “to play recursive function scenarios”.
- Students practice designing and implementing recursive functions.

After the 2-week application of the method, students’ knowledge in recursion was tested (post-test). In the following, we listed the problems the students had to solve:

- Design a recursive function that returns the minimum digit of a given natural number.
- Design a recursive function that returns the number of digit 5 of a given natural number.
- Design a recursive function that returns the minimum value of a given sequence.
- Design a recursive function that returns the number of values 5 of a given sequence.
- Design a recursive procedure that writes the even digits of a given natural number in reverse order and the odd digits from left to right.
- Design a recursive procedure that writes the even values of a given sequence in left-to-right order and the odd values in right-to-left order.

The means of the points the students of the experimental and control groups received in the post-test are 6.75 and 4.75 respectively. Comparing these values with the independent sample t-test, we found that the difference between them is significant ($P = 0.0009 < 0.05$) (favouring the experimental group).

In conclusion, it can be stated that the results of this didactical experiment support our expectation that the multi-sensory method presented in this chapter improves students’ skill to analyse, design, and implement recursive procedures and functions. On the other hand, the modest results of the control group students suggest that teaching recursion “traditionally” at the middle of the first semester is too early. Moreover, even the effectiveness of the multi-sensory method could be higher if it is applied when students are more familiar with the concerned programming concepts.

According to the schema theory, a schema is a cognitive construct that supports learners to memorize, organize, and understand information. It incorporates the characteristics of a number of individual experiences, becoming an abstraction of those experiences. A schema can be seen as a graph, incorporating elements and their interrelationships. The key idea is that once internalized, a schema can be transferred to working memory and processed as a single element. As a result, the cognitive load is greatly reduced (Paas, Renkl, & Sweller, 2003). Moreover, the most useful schemas are triggered automatically (without having to be brought into working memory).

Since beginners do not have such high-level schemas, they are forced to overburden their working memory with multiple lower-level schemas and to consciously apply them (Mead et al., 2006). Since building schemas represents a crucial step in making an expert out of a novice, all teachers must facilitate this

process. In the case of recursive algorithms, the skeletons, their “melody line”, their “staircase representation”, and the “recursive scenarios” can be regarded as algorithmic patterns.

Good writers are also good readers. This seems to be true not only for natural but also for programming languages. Studies like Gray, Edwards, Lewandowski, and Shende’s (2005) conclude that students’ programming skills can be improved by developing their program comprehension abilities. We observed that comprehending the recursive procedures/functions as the ones that are often built around a “key if” instruction and which are characterized by “zones” improves this ability and, consequently, the programming skill.

5 DANCING SORTING ALGORITHMS (STUDY 3)

In order to practice loop instructions and arrays, the majority of computer programming curriculums include sorting algorithms: bubble sort, insertion sort, selection sort, etc. In our third research (Kátai & Tóth, 2010), we analysed how dance can be integrated in the teaching-learning process of the first two sorting algorithms (bubble and insertion sort).

We invited students who like dancing to collaborate in our project. They played the roles of the numbers from the sequence to be sorted. Each of them wore the corresponding number on their dress. The accompanying music pieces were composed on the basis of Michael Flatley's music.

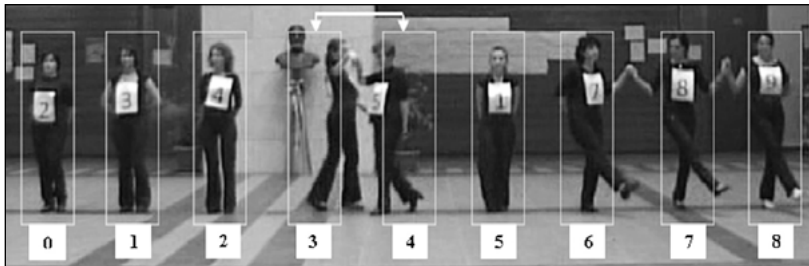


Figure 5.1. *Bubble sort dance performance; dancers 7–9 have already reached their final positions. During the fourth traverse, dancers 5 and 6 perform a changing operation.*

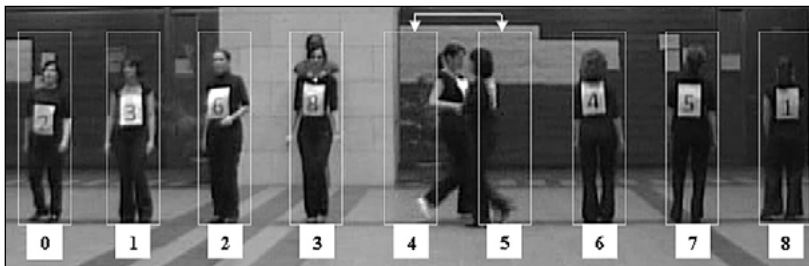


Figure 5.2. *Insert-sort dance performance. The main character of the sixth act of the algorithm, dancer 7, has just begun her insertion process.*

Analysing the above-listed algorithms, we realized that the choreographies have to include two basic elements: 1. comparing two numbers, 2. swapping two numbers. Moreover, bubble sort and insertion sort algorithms can be implemented

in such ways that only neighbouring elements need to be considered. Consulting with the eurhythmics teacher of the faculty, we found that a proper dance step for comparison operations would be that the corresponding two dancers simply turn face-to-face to each other. Regarding the swapping operations, we chose different dance steps adequately to the corresponding dance music. During the performances, dancers that are not in focus are dancing alone, on the spot. In order to make it perceptible that each traverse of the sequence moves at least one number to its final position (in the case of the bubble sort algorithm), dancers who have reached their final place change their dancing style (see *Figure 5.1*). With respect to insertion sort algorithm, at the beginning of the performance, all dancers dance with their back turned to the audience. Each act of the algorithm starts with the next dancer turning to the front. Then, this dancer, as a kind of main character of the current act, inserts him-/herself in the sequence of the already sorted dancers by a comparing-and-swapping sequence (see *Figure 5.2*).

Once these dance performances had been videotaped, we added further graphical elements to the records in order to emphasize that dancer numbers are stored in an array and to highlight the dance couple in the focus. We also included role-playing in the multisensory sorting algorithms lessons. Differences between the stage choreographies and the classroom scenarios were as follows: dance steps were replaced by moves, and the dance music was removed.

5.1 Software tool and suggested syllabus

The multimedia software tool we created helps students analyse the studied sorting algorithms. Some of its characteristics are the following:

- It can be used as both live and self-paced e-learning tool.
- It makes possible backsteps in the interactive animation process. If students miss out on certain details during the stepwise simulation of the algorithm, they can ask the software to step back and repeat the problematic operation.

During sorting algorithm lessons, we applied the following syllabus:

- Teacher presents the strategy the algorithm applies and simulates it on a given number sequence.
- Teacher discusses with students about implementation aspects.
- Students view the dance performance of the algorithm. The teacher, as a narrator, makes references to implementation aspects.
- Students are invited to role-play the algorithm.
- Students simulate the algorithm using the multimedia application referred above.
- Students implement the algorithm.

- As homework, students are asked to work out choreography for selection sort algorithm.

Discussing with the students, they revealed that some of them had the following learning experience:

- After they had attended the explanation and followed the simulation on the number sequence, they thought that they understood how the algorithm works.
- Watching the dance performance, they realized that their understanding of the algorithm had been refined.
- During the role-playing, they observed further details.
- Working at home on selection sort choreography, they understood further subtleties of sorting algorithms.

5.2 The experiment

In order to investigate empirically if dancing and role-playing indeed enhance teaching-learning sorting algorithms, we performed the following experience. Thirty-eight first-year undergraduate students of Sapientia Hungarian University of Transylvania (Târgu-Mureş, Romania) were involved in the experiment. All students started to learn C programming language at the beginning of the 2008/2009 school year.

Before the experiment, we organized a pre-test to form two statistically equivalent groups (experimental/control). In the pre-test, we tested students' knowledge in applying loop instructions. The averages of the pre-test points of the students assigned to the experimental and control groups were 7.06 and 6.4 respectively (according to the 1–10 marking scale in use in Romania). Comparing these performances (independent sample t-test), we did not receive significant differences ($p = 0.35 > 0.05$).

During the experimental period, the 19 members of the experimental group were taught according to the new multi-sensory method. (Since we had proposed to investigate the impact of dance and role-playing on programming education, we excluded from the syllabus the use of the software tool.) In the control group, the 19 students were taught according to classic methods, without making any effort to involve senses in the teaching-learning process.

Once the experiment ended, students' knowledge in sorting algorithms was tested (post-test). In the following, we listed the problems the students had to solve:

- Simulate the two algorithms on a given sequence.
- Implement an insertion sort/bubble sort strategy for sorting in decreasing order.
- Improve the teacher-presented bubble sort algorithm.
- Implement the selection sort strategy.

The averages of the points received for the post-test done by the students of the experimental and control groups were 6.73 and 4.47 respectively. Comparing these values with the independent sample t-test, we found that the difference between them is significant ($p = 0.003 < 0.05$) (favouring the experimental group).

In conclusion, it can be stated that the results of this experiment support our expectation that dancing and role-playing can improve students' skills in analysing, designing, and implementing sorting algorithms.

6 SOME CONCLUSIONS ON OUR MULTI-SENSORY COMPUTER SCIENCE EDUCATION RESEARCH

The presented teaching-learning methods make it possible for students to feel the pulse of algorithms (elementary, sorting, recursive). It is really learning with the whole body. Students can see, hear, and feel (dancing/playing algorithm choreographies/scenarios; drumming algorithm skeletons) the rhythm of algorithms. Having visual/auditory/kinaesthetic experiences of abstract concepts/processes contributes to their deeper understanding. The more senses involved does not only mean more information, better perception, more efficient memorizing, and deeper understanding, but it ensures the same chance for students with different dominant senses.

The high-level right-brain involvement in the teaching-learning process also explains the efficiency of this didactic method. The code-based analysis of the implemented sub-programs takes place mostly on the left side of the brain. By involving the mentioned senses in the learning process, our methods also activate the right side of the brain. With respect to the multimedia applications, during the slow-motion execution/simulation of the algorithms, the instructions of the code are processed as text-based information on the left side of the brain. At the same time, the accompanying music (in the case of recursion) or piano sound sequence (in the case of elementary algorithms) synchronized with the highlight of the current instructions of the code supplies information about the algorithms to the right side of the brain. Through this dual coding, the information is transmitted to the brain in three different forms. On the one hand, it is textual (verbal) and, on the other, it is auditory and visual (non-verbal by images and sensations). While students' ears hear the sounds, the highlight of the current instruction guides their eyes over the running code, so we can say that these three forms bind together in a harmonic way. During the dance performances and algorithm scenes, the textual information is enriched by further visual and kinaesthetic elements. All of these contribute to the effectiveness of the learning process and memorizing.

As the theoretical basis for the effectiveness of the presented teaching-learning strategy, we emphasize the following reasons (detailed in *Chapter 2*):

- Multi-sensory input.
 - Different students – different dominant senses/intelligences/learning style.
- Whole body learning.

- Dual coding (distributed cognitive load).
- Infusion of energy and excitement through music/rhythm/dance/role-playing.

The described experiments represent empirical support for the presented methods.

Consequently, it can be stated that the presented methods and multimedia software tools bring teachers/students closer to Comenius's dream that teaching/learning should be entirely practical, entirely pleasurable, and such as to make school a real game, i.e. a pleasant prelude to our whole life (Dobbie, 1986).

7 THE ALGORYTHMICS LEARNING ENVIRONMENT

In the previous chapters, we described three technologically and artistically enhanced multisensory methods for teaching-learning: (1) elementary algorithms (Kátai, Juhász, & Adorjáni, 2008), (2) recursion (Kátai, 2011), and (3) sorting strategies (Kátai & Tóth, 2010). The artistic elements integrated in the multimedia software tools we developed are the following:

1. The loop skeletons of the elementary algorithms are codified as instrumental sound sequences.
2. Segments from the classical piano masterpiece *Für Elise* by Ludwig von Beethoven are applied as background music for certain instruction zones of the recursive sub-programs during their simulation processes.
3. Videotaped amateur dance performances illustrate sorting strategies.

In recent years, several papers described works that combine science education with art. These initiatives can be categorized as follows:

1. professional art performances that use science as thematic element (art dominates science) (Baldwin & Rivers, 2005; Burg & Lüttringhaus, 2006; Mtangi, 2006; *The dance of mathematics*, 2006; Wechsler, 1997);
2. professional art productions that illustrate abstract scientific models and phenomena (balanced presence of art and science) (Fishwick et al., 2005; Schaffer et al., 2001);
3. art-based methods for science education (science dominates art) (Chavey, 1996; Eisenhower SCIMAST, 1998; Kátai & Tóth, 2010; Moelwyn-Hughes, 2003; Ward et al., 2005).

The advantage of the second category's productions is that they can be exploited from both a scientific and an artistic perspective.

To move our third method (dancing algorithms) into the second category, the AlgoRythmics research group initiated collaboration with a professional art institution (Maros Művészegyüttes), and we replaced modern dance with multicultural Transylvanian folk dances. Romanian, Hungarian, German, and Gipsy folk dance choreographies were designed to illustrate different sorting algorithms. The fruits of this collaboration were such art–science productions which equally promote intercultural and CS education (Kátai & Tóth, 2011).

We decided to design and develop a complex online learning environment at this stage of our research. The basic idea was to supplement the videotaped dance performances with such modules that pave students' way from the comprehension of scientific message of the choreographies to the implementation phase of the programming process.

7.1 The Algorhythmics dance performance collection

As a first step, we selected six sorting algorithms (selection sort, insertion sort, bubble sort, shell sort, merge sort, and quick sort) and six folk dance styles (Romanian – Bihor region; Hungarian – “Székely”; Hungarian – “Csángó”; Hungarian – “Küküllőmenti legényes”; German – Transylvanian Saxon; Gypsy) representing different regions or ethnic groups from Transylvania. Analysing the algorithms, we realized that the choreographies have to include the following basic elements:

- comparing two elements (all algorithms),
- swapping two elements (all algorithms),
- dividing the current sequence in two sub-sequences (quick sort and merge sort),
- partitioning the current sequence in two sub-sequences (quick sort),
- merging two neighbouring sub-sequences (merge sort).

The algorithm–dance associations were established taking into account their common defining characteristics. The number sequence is personified by the dancer sequence (dancers wore the corresponding number on their dresses). Consulting the choreographers, proper dance steps were chosen for all key operations of the algorithms. Almost all choreographies have a dynamic intro simulating the mixing process of the numbers and ending in a vivid finale emphasizing the ordered character of the sequence. The central parts of the choreographies closely follow the corresponding sorting strategies. The balanced art–science collaboration is characterized by the following: (1) The choreographers promote the artistic value of the dance performances (specific dance steps were chosen for swapping two boys, two girls, or a boy and a girl). (2) CS teachers pay attention to the clarity of the scientific message. After the dance performances had been recorded, the videos were supplemented with graphic elements that emphasize that number sequences are stored in one-dimensional arrays.

Figures 7.1–6 present key momentums from our first six algorithmic dance choreographies.



(a)



(b)

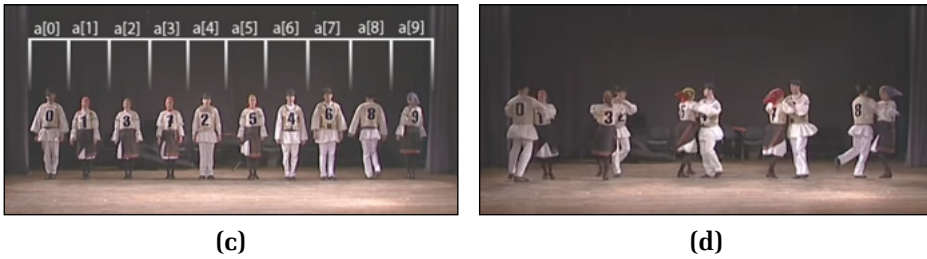


Figure 7.1. Key moments from the Bubble sort *AlgoRythmics* choreography: (a) the sequence to be sorted; (b) a comparing + swapping scene; (c) after the first pass through the sequence (elements that reached their final position turned back); (d) vivid finale emphasizing the ordered character of the sequence



Figure 7.2. Key moments from the Insertion sort *AlgoRythmics* choreography: (a) the sequence to be sorted; (b) the next element ($a[5]$) to be inserted in the ordered left subsequence ($a[0] \dots [4]$) turns to front



Figure 7.3. Key moments from the Selection sort *AlgoRythmics* choreography: (a) the sequence to be sorted; (b) comparing + swapping elements $a[4]$ and $a[7]$ in the selecting process regarding position $a[4]$ (elements $a[0] \dots [3]$ already reached their final positions)



Figure 7.4. Key moments from the Shell sort AlgoRythmics choreography: (a) the sequence to be sorted; (b) comparing elements $a[1]$ and $a[6]$ ($gap = 5$); (c) comparing + swapping elements $a[4]$ and $a[7]$ ($gap = 3$); (d) comparing elements $a[2]$ and $a[3]$ ($gap = 1$)

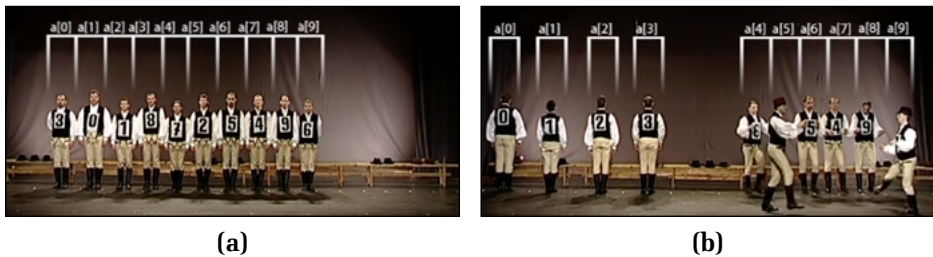


Figure 7.5. Key moments from the Quick sort AlgoRythmics choreography: (a) the sequence to be sorted; (b) comparing + swapping elements $a[5]$ and $a[9]$ in the sorting process of subsequence $a[4] \dots [9]$ (subsequence $a[0] \dots [3]$ is already sorted; current elements are pointed by hats; pivot is in position $a[9]$)



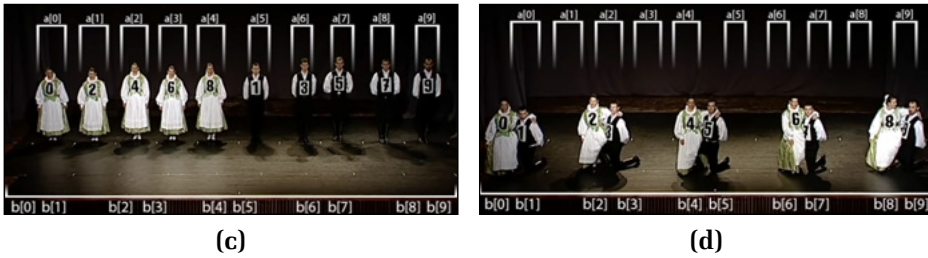


Figure 7.6. Key momentums from the Merge sort *AlgoRythmics* choreography:
 (a) the sequence to be sorted; (b) consecutive dividing scenes;
 (c) subsequences $a[0] \dots [4]$ and $a[5] \dots [9]$ have been sorted;
 (d) subsequences $a[0] \dots [4]$ and $a[5] \dots [9]$ have been merged

7.1.1 Evidence of impact

To test the usefulness of the *AlgoRythmics* dance performance collection, we posted the videos on YouTube and Facebook. In the following, we present extracts from the feedback we received from the international teaching and learning community.

- <https://www.youtube.com/user/AlgoRythmics>
 - 6,638,070 VIEWS, 98.2% LIKES (vs. dislikes), +32.1K SUBSCRIBERS, and 91,272 SHARES.
- <https://www.facebook.com/AlgoRythmics>
 - 3,869 LIKES.

Users' appreciations (selected extracts)¹:

- The words awesome, great, best, nice, brilliant, love, epic, cool, amazing, excellent have appeared hundreds of times in users' comments.
- Best channel ever!
- Best demonstration on the web!
- Thank you for this initiative. This is the greatest piece of art I have ever seen!
- Entertaining and educational at the same time... also I find it incredibly amusing that the 'pointers' are the hats.
- Aha! That would be awesome!!!! BEST IDEA EVER.
- This is the coolest implementation of Quick-sort I've ever seen!
- I will never look at a program the same way again.
- You don't know anything about basic programming until you've seen this.
- Beautiful piece of art to explain computer science, this is so EPIC! Thanks!

¹ The quotations of users' feedbacks are left intact in terms of language, sentence construction, etc. even if they contain some grammatically incorrect elements or misused terms since our aim was to preserve their original wording and style.

- Amazing way to demonstrate the algorithm. Congratulations (many applause)!
- Wow! This combines two of my interests in a way I never expected. Lovely!
- Art + Science = Intelligent entertainment.
- The only true international language!!
- I'm addicted to this video!

Teachers considering AlgoRythmics channel to be useful (selected extracts):

- I really love your work since I'm working with my university to a similar project called Algomotricity.
- I'm a computer science teacher and think that's brilliant! I think I'll give it a try next time I need to teach sort algorithms!
- Boston College CS102 sent me here.
- Awesome. Every computing student and teacher should watch this.
- This is a great way for a beginning CS student to get an intuitive understanding of how quick-sort works. Highly recommended!
- Inspire pour les course!!
- Oh my god! This is so brilliant! Do you have a downlo[d]able version? I'm gonna show this to my students, but internet is not so fancy around here.
- Awesome! I cannot wait to use with my students. I believe it is a great way to learn sorting algorithms. I became a fan of yours.
- Wunderbar! This is science that's joy! I posed the exercise to my students to extract the partitioning algorithm out of the quicksort dance.
- Pushing to have these videos shown in my University's computer science classes.
- Great way to visualize sorting algorithms and useful as an alternative to reading or classroom teaching!
- Awesome!!! I'm programmer and folklore dancer ... and that's what I call interactive learning ... great idea!

Students considering AlgoRythmics channel to be useful (selected extracts):

- It was so hard to learn it only with math and theories. So easy with these videos! Wonderful! Super mega nice work!
- You're awesome! I love it and my professor loves it too!
- Great work! I now understand quick sort. Thanks!
- Epic, I totally understand it now!
- Estos videos son geniales, me salvaron en un examen de programación!
- Never thought that this algorithm is understandable!
- ...I... think I actually understand...
- Man this video helped me more than my teachers ...!!!
- Finally understand the quick-sort method.

- HALLO an meinen Informatik-Grundkurs in Kreuznach!
- I wish my professor were this entertaining.
- This is how our algorithm and data-structure lectures should go like!
- Haha! 30 years after starting university as an IT student I finally understood quick-sort. Great visualization!
- Awesome!!! Before I watched this video I didn't understand this algorithm!
- This is perfect revision as I have my A level computing exam on Thursday.
- We watched this in computer science in school and our teacher said he found this video randomly in the internet.
- This has got to be the greatest math's lesson I've ever had!
- My final exam on trees, graphs, hash tables and all these sort algorithms is Monday, watching these kind of videos is funny and more interesting th[a]n my crappy notebook!
- I saw this in my AP computer science class a couple days ago, and now I'm hooked on this song.
- Finally someone explains it in such a way I can understand!
- We are watching select-sort right now in class. Awesome!
- Brilliant! The bubble sort dance illustrates the concept so well that our teacher shows it in class!
- If my professors did this diddy for class I would not only be amused but I'll never forget. New teaching technique? I sure hope so!
- You can visually see the Algorithm in execution. Great!
- When I have to explain quick sort in my oral exam I may just dance ...

Extracts from online articles that appreciated AlgoRythmics channel (selected extracts):

- (<http://www.i-programmer.info>) You may well have seen many simulations of sorting algorithms that aim to show in novel ways how the algorithm works or perhaps doesn't work quite as well as it should. However I guarantee that you have never seen anything quite in the same league as the videos made by Sapientia University – they are simply crazy but in the nicest possible way. ...

If you have been following the surreal interpretations of sorting algorithms as folk dances here is the ingenious culmination – the Quicksort, the most difficult of algorithms, complete with hats as pointers. Yes I know I claimed that it would be impossible, or if possible the result would be a modern dance the like of which we have not seen, but... they have done it. The slightly insane dance group at Sapientia University has put together a Hungarian folk dance with steps that follow the Quicksort algorithm. It is worth noting that it takes just short of 7 minutes to sort just ten dancers which really isn't very quick; that only males take part which proves that it is a very dangerous

algorithm and, oh yes, two hats are used to mark the progress of the scan! Clever stuff! Now see if you are anywhere near as clever by verifying that they are in fact dancing the Quicksort. If you need help then all I can suggest is that you keep your eye on the hats, notice exactly what happens when they meet and pay attention to the partitions that are produced. The sort of the dance has now reached Merge Sort and it's very tricky – how does it end up with the boy-girl pairing? ...

- (<http://www.dyxuchen.com>) The next time you're having trouble in computer science classes grasping the nature of shell sorts and bubble sorts, head over to the AlgoRythmics YouTube channel. There, a group of folk dancers are using Hungarian and Gypsy folk dance to teach people exactly how those sorting operations work, complete with graphic overlays and numbers on their chest and back to show you how the sort functions. ... The whole affair is part of Sapientia University's Intercultural Computer Science Education program. If you understand the sorting algorithms that are on display, you'll really appreciate them. If you don't understand them but need to learn, you'll like them a lot. If you have no idea what's going on, at least you'll enjoy the dancing.
- (<http://mrhodotnet.blogspot.com>) Absolutely brilliant series of videos interpreting the various sorting algorithms using folk dance. Nice touch on the conventions for comparisons. The videos have no voice-over or distracting subtitles and are slow enough that you can show the video and comment on each of the steps in the algorithm while it is occurring. After a few steps, have students predict what will happen next (e.g. who will dance next and who will swap places).
- (<http://antoniofarinha.com>) Computer-related algorithms can be quite hard to understand, and in some cases a simple visual demonstration makes it so much easier. The IT people in the audience might have seen a few of them, but I'm pretty sure none was quite as peculiar as this. AlgoRythmics, a folk-dancing group from Romania's Sapientia University, decided to show how data sorting algorithms work by showing them as dances. Bizarre, right? But strangely mesmerizing, and it sure does the trick of explaining how they work. Until now they did insert-sort, bubble-sort, select-sort and shell-sort. I hear that merge-sort and quick-sort are coming soon.

7.2 AlgoRythmics animations

The term “algorithm visualization” (AV) technology usually refers to video or computer-based animations that visually illustrate how algorithms work (Shaffer et al., 2010). A recent meta-analysis comparing instructional

animation with static pictures confirmed the educational effectiveness of representational animations, especially when procedural-motor knowledge had to be assimilated (Höffler & Leutner, 2007). The meta-study of AV effectiveness presented by Hundhausen, Douglas, and Stasko (2002) concluded that AVs foster effective learning when students are actively involved in the visualization instead of passively viewing it. For example, students should be invited to run the animation for several inputs observing the input variant/invariant characteristics of the studied algorithm.

A genuinely active involvement offers learners the opportunity to have control over the algorithm animation process, even orchestrating it (Mayer & Chandler, 2001). Algorithm orchestration is a special case of the so-called interactive prediction method (Hansen, Narayanan, & Schrimpscher, 2000), when students are invited to predict and even perform (using an interactive visual learning environment) the entire step sequence of the algorithm. Features such as immediate feedback, possibility to try again and help button (available at each step of the algorithm) can guarantee that they will be able to complete their task.

Accordingly, we have decided to attach to each dance choreography an abstract computer animation. The array that stores the numbers to be sorted is visualized as a white-box sequence (numbers are visible). Comparing/swapping two elements is animated by proportionally expanding/interchanging the corresponding boxes. Pairs of arrows direct the user's attention to the elements the current operation is applied to. Elements that have reached their final positions in the sorted sequence are recoloured.

In line with the principle of genuine active involvement, the animation module operates in two interactive modes:

- White-box task: The user is invited to predict the compare/swap operation sequence of the studied sorting algorithm for a randomly generated number sequence stored in a white array (numbers are visible). In order to implement the predicted next operation, one needs to click on the 1. corresponding element pair (parameters), 2. the appropriate action button (compare/swap). Possible immediate feedbacks (in this order):
 - wrong parameter number,
 - wrong parameters selected,
 - wrong action selected,
 - the software animates the correctly predicted operation (by proportionally expanding/interchanging the corresponding boxes).

To prevent a “getting stuck experience”, we implemented a help button to inform users about the next correct operation.

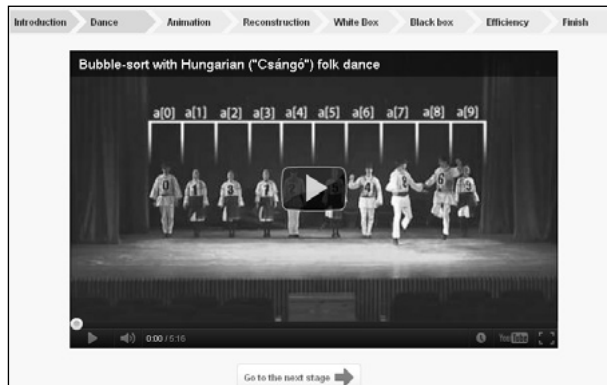
- Black-box task: The user has to perform the same task but on a random sequence stored in a “black array” (numbers are hidden). After the user has selected correctly the next element pair and pushed the proper

operation button, the software applies the corresponding animation. Although the numbers are invisible, users can realize the result of the currently performed (and animated) comparison since the corresponding boxes are expanded proportionally to the values they are storing.

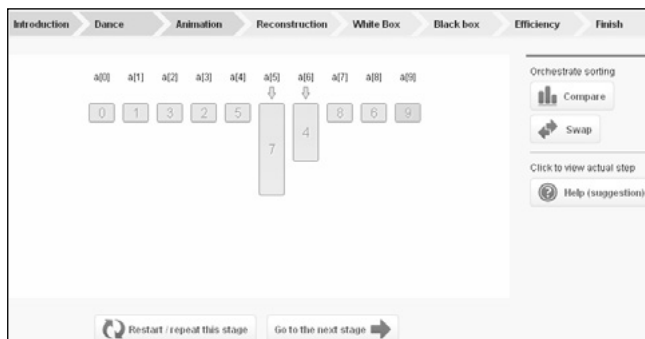
The environment also integrates an algorithm comparison module. The sequence to be sorted is the full colour scale. The simulation starts with parallel identical colour arrays generated randomly. Students can watch the parallel slow-motion execution of several sorting algorithms. The basic differences between the strategies the algorithms apply become palpable in a specific way.

Figure 7.7 presents the six learning steps included in the first version of the AlgoRythmics environment (see the menu bar).

In the next two chapters, we present studies that examined (1) the algorithmic folkdance choreographies from the perspective of inter- and multicultural CS education and (2) the effectiveness of the two operating modes (white/black box) the attached computer animations incorporate.



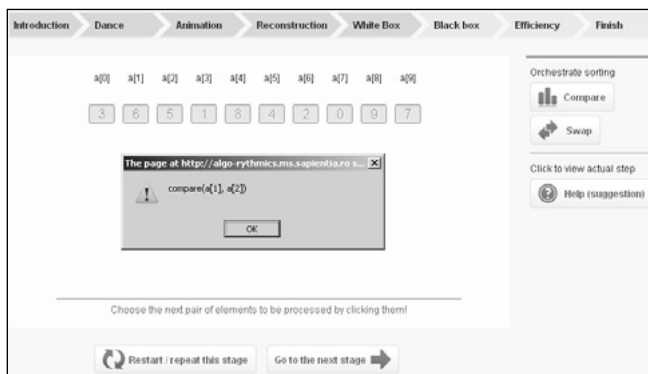
(a)



(b)



(c)



(d)



(e)



(f)

Figure 7.7. Learning steps in the AlgoRhythmics environment: (a) dance choreography illustration; (b) abstract animation; (c) interactive reconstruction of the algorithm (feedback: wrong action selected); (d) orchestrating the algorithm on a sequence stored in a white array (help request); (e) orchestrating the algorithm on a sequence stored in a black array; (f) different sorting algorithms are visualized as they are working side by side on different colour scale bars

8 ALGORYTHMICS: SCIENCE AND ART WITHOUT ETHNIC BORDERS (STUDY 4)

Undoubtedly, the issues of Computer Science Education (CSE) are of paramount importance in the twenty-first century. What is another critically important educational topic? The issue of inter- and multicultural education (Banks & Banks, 2001). According to a recent report of the Committee on Culture, Science and Education (C-CSE, 2009), the Parliamentary Assembly of the Council of Europe recalls that education shall promote understanding, tolerance, and friendship among nations and ethnic groups, and all forms of artistic expression are tools in intercultural education (IcE). Being motivated by these trends, we have proposed to try to combine these two topics.

As detailed in the previous chapter, we particularly focused on multicultural content integration in CSE through art-based pedagogical tools. We proposed to design an online e-learning environment that has the potential to equally promote both IcE and CSE. The folkdance choreographies we created illustrate basic CS concepts on the one hand and the cultural diversity of Transylvania (Romania) on the other. This unique art–science combination also illustrates how the concept of “unity-in-diversity” can be implemented in a science educational context: (1) multicultural artistic performances to promote the cause of universal science; (2) scientific content in an artistic framework.

The study we performed (Kátai, 2014b) revealed possible difficulties CS teachers may face when they are presenting scientific content in culturally diverse contexts (especially in regions with cultural tension). Research results revealed that students’ culture-related concepts and feelings may even influence the way they relate to the scientific content.

8.1 Artistically enhanced multicultural education

The last few decades have seen considerable efforts on the part of scholars and policy makers to embark on initiatives to acknowledge, accept, and value cultural diversity in place of the accustomed melting pot approach, whose objectives have been to assimilate minorities into the mainstream at the expense of their cultural identities. Amaram (2007) identifies three organizational dynamics that contribute to the growth of the diversity perspective:

- Social justice (moral, ethical, and social responsibilities towards minority members of society).

- Legal obligations to eliminate racial and ethnic discrimination in education and employment.
- Globalization, with its multicultural implications, has become an indispensable factor for business organizations in strategic competitiveness.

According to the above referred report of the Committee on Culture, Science and Education (2009), the society's need for the particular competences and qualities that are developed through artistic and cultural education is greater than ever. C-CSE defines cultural education as: (1) learning and practising the arts; (2) learning through the arts (which means the use of art-based forms of teaching as a pedagogic tool in all kinds of school subjects); (3) using the arts for the promotion of cultural and social objectives such as mutual respect, understanding and tolerance, appreciation of diversity, team work, creativity, etc.

Different words like "multicultural" and "intercultural" have been used to describe the changes that have been taking place in modern society. These terms, on the one hand, describe a society in which different cultures live side by side and, on the other hand, express the conviction that we all become more by coming into contact with and experiencing other cultures and that people of different cultures can and should be able to cooperate with each other and learn from each other (National Council for Curriculum and Assessment, 2005).

Valdez (1999) concludes that multicultural education has the potential to decrease race, ethnicity, class, and gender divisions. Gay (2000) emphasizes that it is imperative that teachers learn how to recognize, honour, and incorporate the cultural characteristics of students into their teaching strategies. Although definitions of multicultural education may vary (Banks, 1977; Banks & Banks, 2001; Baptiste, 1979; Bennett, 1990; Frazier, 1977; Grant, 1977; Hunter, 1974; Nieto, 1992; Sizemore, 1981), most multiculturalists agree that multicultural education means learning about, preparing for, and celebrating cultural diversity, and it requires changes in school programmes, policies, and practices (Gay, 1994).

With respect to the role of arts in intercultural education, the C-CSE (2009) report states that:

- Art can efficiently reinforce formal education. Cultural and artistic means of education should become an important part of school curricula.
- Diversity and a multicultural environment stimulate creativity.
- Intercultural dialogue is the basis for harmonious and peaceful co-existence.
- Music, art, and dance can be effective tools for intercultural education.
- Educational institutions should initiate international co-operation projects in cultural education, especially in regions with political (or ethnic) tensions.
- Art and educational institutions need to rethink their roles in connection with cultural education. A new learning culture has to be promoted by enabling new learning communities and supporting networks.

- Recent cultural monitoring studies have shown that parents would like to see more art and culture in schools because they believe that cultural education plays a very important role in the comprehensive development of their children's personalities.

Oreck's (2004) study reveals that awareness of student diversity and the need for improved motivation and enjoyment in learning are the dominant teacher motivations for using arts. Field (2010) examined how arts education, and specifically music, in the International Baccalaureate Middle Years Programme promote intercultural awareness. He concludes that an international curriculum should aim for intercultural/international understanding. According to Thomas and Mulvey (2008), arts promote student understandings of the values, goals, and practices of community-based work and enable meaningful student roles in community-based partnerships.

New information and communication technologies have strongly increased the possibilities for and the impact of cultural education both in formal and informal education. According to Gadsden (2008), the study of the arts (music, visual art, performance, etc.) in education has taken on new venues in supporting learning and teaching through technology and multimedia.

Despite these clear directives, very little is being done to put these insights into practice (C-CSE, 2009). Critical pedagogy theorists identify the gap between theory and practice as a major weakness of multicultural education (May, 1994; Wilhelm, 1994). Regarding arts as key tools in intercultural education, data collected from 423 K–12 teachers (enrolled in professional development programmes for general education teachers in the United States) indicate that teachers believe the arts are important in education but rarely use them (Oreck, 2004).

According to Hoffman (1996), multicultural projects often tend to concentrate on cultural celebrations on special occasions or dates and thus are implemented at the superficial level of food, dance, and music. Multicultural education is frequently considered as an appendix to the regular curriculum. As a result, the practice of multicultural education is minimized to folklorization.

With respect to cross-curricular links between arts and other subjects, the Education, Audiovisual and Culture Executive Agency of the European Commission (2009) reports that:

- Just over a third of European countries establish cross-curricular links between arts and other subjects at a curriculum level, either through educational objectives or through subject-specific links.
- In some cases, promoting cross-curricular links is explicitly stated as an aim/objective of the arts curriculum. (Only one country has subject links between the arts and sciences.)
- In several countries (including Romania), cross-curricular links between arts and other subjects may be established at a local or school level.

Accordingly, effective multicultural education means that cultural pluralism permeates all dimensions (including the curriculums in all subjects at all levels) of the educational process. According to Gay (1994), advocates of multicultural education suggest three general approaches for how it can be accomplished in school practice: (1) teaching content about cultural pluralism, (2) teaching culturally different students, and (3) using cultural pluralism to teach other academic subjects and intellectual skills. Most multiculturalists agree that the specific content, structures, and practices employed in achieving multicultural education should differ depending on the setting. It is expedient for teachers to elaborate their own definitions of multicultural education adequate to their specific needs, rather than imposing a rigid all-embracing structure to it (Gay, 1994). Consequently, although effective multicultural education presupposes institutional top-down strategies, educators should take the initiative to implement principles of multicultural education in their own courses, rather than waiting for lagging organizational solutions.

Although there are more opportunities for teachers to use ethnic and cultural content to illustrate concepts, themes, and principles in the social studies, the language arts, and in music, multicultural education also provides a perspective for maths and science. For example, ethno-mathematics (Nelson-Barber & Estrin, 1995; Tate, 1995) presents a view of mathematical thinking that incorporates the ways in which culture and mathematics are intertwined. In the sciences, there is the opportunity to study environments from the perspectives of the diversity of cultural knowledge (Harding, 1998; Sleeter, 1996). Another possibility is to use multicultural art-based pedagogical tools in teaching-learning scientific subjects. The best methods in this category equally promote multiculturalism and science education.

8.2 Intercultural computer science education in Transylvania

In this study, we examined whether students' ethnic background might influence (or perturb) their approach to the scientific content in terms of the cultural frame (context) in which it is presented. We designed our study taking into account that, in order to be effective, science education must consider the cultural context of the society which provides its setting and whose needs it exists to serve (Wilson, 1981). This is especially true for intercultural science education and also for IcCSE.

Transylvania is a pronouncedly multicultural part of Romania with several ethnic groups having coexisted in this region for centuries (Romanian, Hungarian, German, Gipsy, etc.). The most numerous minority is the Hungarian

one, and since (during its controversial history) Transylvania has often been the scene of Romanian–Hungarian political tensions, this chapter refers to these two cultures as “opposite ones”.

Educational institutions can be classified as mono-, bi-, or multicultural ones. Promoting IcE in such a specific school environment, and in a neighbourhood with similarly diverse cultural characteristics, may imply particular challenges to be taken into account. Accordingly, the high schools selected for involvement in the experiment were: Schools_1 (Romanian mono-cultural institutions), Schools_2 (Hungarian mono-cultural institutions), and School_3 (bi-cultural institution with both Romanian and Hungarian educational programmes).

Our core research question was: How might the existing cultural concepts held by students (from mono- and bi-cultural institutions) affect their appreciation of sorting algorithms presented in a cultural context different to their own?

Ninety-six 9th grade novice CS students participated in the experiment from three secondary education institutions: School_1 (31 Romanian students, group monoRO), School_2 (21 Hungarian students, group monoHU), and School_3 (18 Romanian students, group biRO; 26 Hungarian students, group biHU). We performed the experiment after students had been introduced to programming but before they had studied sorting algorithms. The concept of algorithm complexity (efficiency) was also introduced with respect to sorting algorithms. The age of the students was around 15 to 16 years. The percentage of female students was 27%.

The experiment was conducted in computer labs, where all students had individual access to the online e-learning tool. Before commencing the e-learning session, students were briefly introduced to or reminded of the concepts of algorithms and sorting algorithms. In addition, a brief overview of the e-learning session was presented to them. The students were asked to utilize a 7-point scale (ranging from strongly disagree to strongly agree) and were invited:

- to watch Selection sort with GIPSY folk dance (see *Figure 7.3*);
- (item 1a) to respond to the sentence: “I liked this algorithm dance show” (1: Strongly disagree ... 7: Strongly agree);
- to watch Insertion sort with ROMANIAN (Bihar region) folk dance (see *Figure 7.2*);
- (item 1b) to respond to the sentence: “I liked this algorithm dance show” (1: Strongly disagree ... 7: Strongly agree);
- to watch Bubble sort with HUNGARIAN (“Gyimesi csángó”) folk dance (see *Figure 7.1*);
- (item 1c) to respond to the sentence: “I liked this algorithm dance show” (1: Strongly disagree ... 7: Strongly agree);
- (item 2a) to answer the question: “Which algorithm performed the sorting task most efficiently? GIPSY/ROMANIAN/HUNGARIAN”;
- (item 2b) to answer the question: “Which algorithm performed the sorting task most efficiently? Bubble sort/Selection sort/Insertion sort”;

- (item 3) to choose the algorithm they found easiest to understand: “Selection sort with GIPSY folk dance/Insertion sort with ROMANIAN folk dance/Bubble sort with HUNGARIAN folk dance”;
- to watch a bubble sort algorithm animated on a white-box array (the numbers being visible); all students faced the bubble sort algorithm, but without being informed of this fact (see *Figure 7.7.c*);
- to orchestrate the bubble sort algorithm on a randomly generated sequence stored in a white-box array (see *Figure 7.7.d*);
- to orchestrate the algorithm on a randomly generated sequence stored in a black-box array (being informed about the results of the comparison operations) (see *Figure 7.7.e*).

8.2.1 Results

Table 8.1 presents the means of students’ response scores with respect to the three algorithm dance shows. It can be seen that both Romanian (RO = monoRO + biRO) and Hungarian (HU = monoHU + biHU) students responded most favourably to the performance that represented their own culture. They showed the following preferences:

- Romanian students: 1) RO, 2) Gipsy, 3) HU;
- Hungarian students: 1) HU, 2) Gipsy, 3) RO;

Table 8.1. *Response scores with respect to the algorithm dance shows*

	Gipsy	Romanian	Hungarian
RO	5.24	5.65	5.16
HU	5.45	5.40	5.96
monoRO	5.23	5.55	5.58
monoHU	5.10	5.43	5.43
biRO	5.28	5.83	4.44
biHU	5.73	5.38	6.38

Source: based on author’s measurements

Comparing students’ response scores with respect to their own and to the opposite culture (paired sample t-test), we found that Romanian (RO) students yielded a value of $p = 0.09$ (not significant) and Hungarian (HU) students: $p = 0.009$ (significant). No significant differences were detected between Romanian and Hungarian students’ responses to the Gipsy dance performance.

Analysing students’ scores at the group level, we observed that in the bi-cultural institution (School_3) both Romanian and Hungarian classes appreciated

the performance that represented their own culture significantly more than the opposite one (biRO: $p = 0.04$; biHU: $p = 0.0003$). In the case of mono-cultural institutions, we detected equality (School_2) or even a better (School_1) appreciation (not significantly) of the opposite culture.

More important are the results regarding algorithm efficiency when algorithms were identified by the corresponding culture (item 2a). We coded the students' choices as follows:

- 1: if they chose their “own folk dance choreography” as the most efficient one,
- -1: if they chose the “opposite folk dance choreography” as the most efficient one,
- 0: if they chose the “neutral folk dance choreography” (Gipsy) as the most efficient one.

Again, while in the case of groups monoRO and monoHU the means were equal to zero (0: 11 (own), 11 (opposite), 9 (neutral)) or near to zero (-0.04: 9 (own), 10 (opposite), 2 (neutral)), these average values for the groups from the bi-cultural institution were 0.72 (biRO; 14 (own), 1 (opposite), 3 (neutral)), and 0.42 (biHU; 16 (own), 5 (opposite), 5 (neutral)). In the bi-cultural institution (School_3), significantly more students chose the algorithm that was presented in their own cultural context as “the best one” than in the mono-cultural ones (School_1 + School_2) ($p = 0.006$; chi-square test). At the group level, we observed the following results: 0.01 (monoRO–biRO), 0.1 (monoHU–biHU).

A comparative analysis of the answers to items 2a and 2b also revealed that students from School_3 were “too much focused” on the cultural aspects. In the case of groups biRO and biHU, only 44.44% and 38.46% of the members, respectively, chose the same algorithm when these were identified by culture/strategy (items 2a/2b). These percentages for the monoRO and monoHU groups were 64.52 and 61.9 respectively. By applying independent sample t-test, we observed p-values close to the limit of significance: 0.089 (biRO–monoRO), 0.057 (biHU–monoHU).

Applying the same (1, -1, 0) coding to students' responses to item3, we observed quite similar results to those of item 2a:

- means near to zero for groups monoRO (-0.09: 11 (own), 14 (opposite), 6 (neutral)) and monoHU (-0.04: 7 (own), 8 (opposite), 6 (neutral));
- positive means for groups biRO (0.61: 12 (own), 1 (opposite), 5 (neutral)) and biHU (0.19: 12 (own), 7 (opposite), 7 (neutral));
- significant p-value when we compared biRO + biHU with monoRO + monoHU (0.03).
- at group level: 0.01 (monoRO–biRO), 0.62 (monoHU–biHU).

A limitation of this study is that the students could be misled by the lengths of the video recordings (see *Table 8.2*).

Table 8.2. *Some data about the operation sequences and the video recordings*

	Elementary operations (Comparing + Swapping)	Lengths of the whole dance performances (sec)	Lengths of the sorting processes (sec)
Insertion sort	37 (23+17)	4.04	2.25
Bubble sort	39 (26+13)	5.15	3.00
Selection sort	59 (45+14)	7.06	6.20

Source: based on author's measurements

Comparing students' performance results (number of errors and help requests) with respect to their white- and black-box task (mono-bi; monoRO-biRO; monoHU-biHU; boys-girls), we did not find any significant differences.

8.3 Conclusions

Interestingly, Filpisan, Tomuletiu, Gyorgy, and Moldovan (2012) conducted a parallel research (independently) in a related topic in the same town from Transylvania, Romania. They implemented an intercultural educational project, addressing teenagers only in School_1 and School_2 (lessons with intercultural concepts as the topic; workshops on the subjects of: defining interculturality, culture profiles of Romanians and Hungarians, prejudices and discrimination). 100 ninth-grade (four classes) Romanian and 100 ninth-grade (four classes) Hungarian students were involved in the experiment. One of the purposes of the project was to give a "wake-up call" to teachers regarding the importance of "educating for interculturality". The authors conclude that one of the educational goals of Transylvanian educational institutions should be to prepare students as citizens of an extended multicultural society such as the European Union or our globalized multicultural world. Our conclusions harmonize with this "wake-up call" too.

The study we performed revealed possible difficulties CS teachers may face when they are presenting scientific content in culturally diverse contexts (especially in regions of cultural tension). Research results reveal that students' culture-related concepts and feelings may even influence the way they relate to the scientific content. While we expected that students' responses would show higher appreciation of their own culture-related dance performance, it was surprising to notice that in the bi-cultural institution they made a similar choice regarding the most efficient algorithm.

Bennett distinguishes between ethnocentrism and ethnorelativism. Ethnocentric people do not have internalized perspectives emanating from other cultures and tend to value their own culture above everything else. Their early training in

the home (and sometimes in the school) creates the habits of mind that characterize them. Conversely, ethnorelative people appreciate cultural perspectives other than their own and recognize that particular cultures can only be understood within a cultural context (Bennett, 1993).

According to our findings, the bi-cultural character of an educational institution does not promote ethnorelativism implicitly. What is more, it can polarize differences. On the other hand, the mono-cultural character of an educational institution does not promote ethnocentrism implicitly either. Bennett describes a six-stage process that moves someone from ethnocentrism to ethnorelativism. Carefully designed IcCSE could have the potential to become a promoter of this process.

9 TEACHING “NOT BLIND LEARNERS” TO PROGRAM “BLIND COMPUTERS” (STUDY 5)

Computer algorithms are inherently abstract entities. Since they lack any tangible real-world representation, it is quite difficult to teach and learn them. As Turing (1950) stated: “One’s object is then to have a clear mental picture of the state of the machine at each moment in the computation. This object can only be achieved with a struggle.” Graphical representation of an expert’s “clear mental picture” of the algorithm has become the most common method of illustrating how algorithms work. According to a recent survey, “while many good algorithm visualizations are available, the need for more and higher quality visualizations continues” (Shaffer, Cooper, & Edwards, 2007; Shaffer et al., 2010).

Compared to humans, computers are blind in many ways. An algorithm visualization (AV) system may visualize information that has extra meanings for learners. In such enriched learning environments, learners may find it difficult to follow a strict computer algorithm. For example, sorting AVs usually expose the number sequence to be sorted. Since learners see the numbers, they implicitly realize if two elements are in the right order or not and may skip the explicit comparison operation of the computer algorithm. In the present study (Kátai, 2014a), we investigate this phenomenon and suggest solutions to avoid potential side-effects of substituting “blind computers” with “not blind humans” in AV learning environments.

9.1 Theoretical background

Theories of Epistemic Fidelity and Cognitive Constructivism provide the theoretical framework for this study. Wenger (1987) defined the epistemic fidelity of a representation as the degree to which an external representation of a phenomenon reflects the expert’s model of this phenomenon. Epistemic Fidelity theory has its origin in representationalist epistemology (Newell, 1980), which assumes that objects can be represented in the mind by symbolic models, and these “images” are the basis for human reasoning and action (knowledge representation assumption). According to Hundhausen (1999), other assumptions of the epistemic fidelity view are:

- Knowledge flow assumption: (1) Transmitter’s knowledge is encoded in a graphical representation; (2) Graphical representation is decoded by the receiver. In terms of AV: knowledge is seen to flow from teacher to AV to student through the conduit of the visual medium.

- Graphical medium effectiveness assumption: Graphical representations are effective tools for presenting mental models. They have the potential to support representations that closely match the source mental model at an appropriate level of abstraction (eliminating unnecessary detail).
- Epistemic Fidelity Assumption: High-epistemic-fidelity encoding promotes efficient decoding.

Hand in hand with IT advances and based on the epistemic premise that more encoded information results in more decoded information, AV researchers have developed more and more sophisticated AV software tools. On the other hand, recent research results have revealed other factors in addition to epistemic fidelity, contributing to successful knowledge transmission. One of them is the learner’s level of attention: heightened learner attention results in more robust and efficient decoding (Lawrence, 1993). In addition, heightened attention can be fostered by increasing the learner’s level of involvement. Students’ active engagement in the learning process is even more important from the perspective of the Cognitive Constructivism theory.

According to the Cognitive Constructivism theory, meaningful learning involves active knowledge structure construction (Carey, 1985). Students filter and interpret (based on their prior knowledge) any new information, and this process results in progressively reconstructed conceptual schemes (Driver, 1989). Active knowledge construction implies the active use of new information by applying it to new situations (Mayer, 1999). The role of technology is not simply to transmit knowledge but to support the knowledge construction process. Accordingly, in order to benefit most from AV technology (no matter how high the level of its epistemic fidelity), students have to be meaningfully involved in the algorithm visualization process (Hundhausen et al., 2002). For example, environments which engage students in so-called interactive prediction may promote more effective learning (Hansen et al., 2000).

Interactive prediction basically means that the animation process is interrupted and the viewer is invited to predict what the visualization will show next about the algorithm. This feature can be implemented in several ways:

- Students can be invited to orally predict what would happen at pre-defined breakpoints in the visualization processes. According to Byrne, Catrambone, and Stasko (1996), these “interventions” in AV processes led to significantly better post-test results.
- In Korhonen and Malmi’s (2000) study, students had to manually trace the execution of algorithms. In addition, in some key points of the algorithm, students were invited to record (using a graphical editor) their answers to questions about the current state of the data structure (and they received immediate feedback about their responses). The authors reported excellent results with their AV system.

- Naps, Eagan, and Norton (2000) also reported anecdotal success with the incorporation of stop-and-think questions into their AV system. They found that forcing students to answer the questions, registering their responses for grading purposes, and giving them immediate feedback could result in more effective learning.
- On the other hand, Jarc, Feldman, and Heller (2000) reported opposite results. The AV system they developed presents students with algorithm animations in two modes: (1) “Show Me” (students passively watch the animation, trying to learn the behaviour of the studied algorithm); (2) “I’ll Try” (Students are engaged with interactive prediction questions). Surprisingly, the group that used the interactive prediction feature of the AV system performed worse (but not significantly) than the group that did not use it.

Possible enhancing/diminishing factors contributing to the above results are:

- The immediate feedback that students receive after they have answered the questions can reset confused ones’ perception of the algorithm back to the track intended by the teacher (Naps et al., 2000).
- Instead of thinking thoroughly on the questions, students (especially weaker ones) may tend to view interactive prediction as a guessing game (Jarc et al., 2000).

In the following paragraphs, we analyse further factors that could increase/diminish the effectiveness of AV systems that include interactive prediction.

9.2 Student-orchestrated computer algorithms

A special case of interactive prediction is when students have to orchestrate the studied algorithm. They are invited to predict and even “perform” (for a given input; using an interactive visual learning environment) the entire step sequence of the algorithm. Features like immediate feedback, possibility to try again, and help button (available at each step of the algorithm) can guarantee that all students will be able to complete their task. The primary goal of the orchestrating process is not to assess but to enhance or refine students’ understanding about the studied algorithm. Obviously, this “You Are in Charge” phase of the learning process should be preceded by “Preparation” phases (teacher explanation; watching the animated algorithm), when students are initiated into and familiarized with the strategy the algorithm is built on. A possible scenario could be: “Listen to It” (to be initiated into) + “Watch It” (to become familiar with) + “Try It” (to assimilate it).

In such learning environments (especially during “Try It” phases), users become active players of the AV process. If an AV system has this feature, then the concept of epistemic fidelity acquires new connotations. “Not blind users”

are invited to process algorithms (in terms of their high-level operations) created to be processed by “blind computers”. Effective AV systems should support users in identifying with the computer. From this perspective, high epistemic fidelity implies that at each state of the algorithm animation process users are not more informed (they do not see more) than the computer would be. For example, as we mentioned above, almost all computer algorithms include comparison operations (depending on how the compared values relate to each other, the computer chooses between parallel scenarios to be followed). If the AV software shows the values to be compared, students may perform the comparison in their mind, implicitly (without realizing it). As a result, they may skip the explicit comparison operation of the algorithm. On the other hand, by hiding the values to be compared, AV designers can force students to explicitly perform the comparison operation (as part of the animation process). Subsequently, students are informed about the result of the comparison by the AV software.

We have proposed to investigate the following questions:

- Could it happen that information that might have extra meanings for human viewers obstructs them in following strict computer algorithms?
- Can wisely applied hiding result in more effective AV?

9.3 Selective hiding for improved algorithm visualization

Sorting algorithms are probably the most directly perceived computer algorithms by IT users. Sorting a list is a common operation in many fields of work and is one of the most fundamental problems in CS. Bubble sort is a popular introductory sorting algorithm that works by repeatedly stepping through the list to be sorted, comparing each pair of neighbouring elements and swapping them if they are in the wrong order. The pass through the list is repeated until no swaps are needed, which indicates that the list is sorted. The algorithm can be easily “optimized” by observing that:

- The i^{th} pass is guaranteed to put the i^{th} largest element into its final place.
- All elements after the last swap of the current pass are in the right order, and they do not need to be checked again in the next pass.

As we described in the *Chapter 7*, the AlgoRythmics environment includes two sorting algorithm orchestration tasks on sequences stored in white or black arrays (the numbers are visible/invisible). By inviting students to participate in these interactive processes, we managed to generate the so-called “not-blind learners processing blind computer algorithms” phenomenon. We expected that students performing the white-box task might have problems with comparing operations. We anticipated that they might tend to:

- skip the explicit comparing operation if the current element pair had to be swapped,
- omit to deal with the next element pair if these are in the right order.

Our particular research question was: Can the “comparison problem” (resulting from replacing blind computers with not-blind learners) be diminished by hiding the numbers to be sorted (resulting in increased epistemic fidelity)?

9.4 The experiment

We initially proposed to involve in the experiment (presented to the students as a mandatory testing) all 161 first-year (school year 2013–2014) undergraduate students enrolled in sciences-oriented programmes (Sapientia University: Computer Sciences, Electrical Engineering and Mechanical Engineering). Since the group of our first-year students is consistently heterogeneous (regarding their prior knowledge in programming), we organized a pre-test to create two statistically equivalent groups (experimental/control). One hundred and fifty-one students participated in the pre-test. They had to solve the following problems:

1. Elaborate an algorithm to determine the “minimax points” of a numerical input matrix (to detect students with above average programming skills).
2. Elaborate an algorithm to compute the averages of the negative/positive elements of a numerical input sequence (to detect students with average programming skills).
3. Imagine n jail-cells ($1\dots n$). Initially all cells are locked. A jailer traverses the cells n times. During his i^{th} ($i=1\dots n$) traverse, s/he changes the status (locked/unlocked) of door j ($j=1\dots n$) if i divides j . What is the number of doors remaining unlocked after the jailer has finished his/her n^{th} traverse? (a) $n = 10$; (b) $n = 1,000$; (c) give a formula for the general case (to detect students without prior programming knowledge but with promising algorithmic thinking).

We characterized students’ programming skills as follows:

- “Above average”: $\text{task_1} \geq 50\%$;
- “Average”: $\text{task_1} < 50\%$, $\text{task_2} \geq 50\%$;
- “Promising”: $\text{task_1} < 50\%$, $\text{task_2} < 50\%$, $\text{task_3} \geq 50\%$;
- “Others”: $\text{task_1} < 50\%$, $\text{task_2} < 50\%$, $\text{task_3} < 50\%$.

Based on their pre-test results, we divided students into two groups:

- A: control group (76 members; average score: 4.06),
- B: experimental group (75 members; average score: 4.06).

Obviously, no significant differences were detected between groups A and B: $p = 0.49 > 0.05$ (independent sample t-test).

As usual, not everything went exactly as we had planned. Only 138 students participated in the e-learning experiment. Furthermore, fifteen of them had to be eliminated because they had not accomplished their white-/black-box task.

In addition, some students from group A forgot to indicate their group ID (at the registration) and the software assigned them implicitly to group B. In the end, we had a valid control group with 52 members (group vA) and an experimental one with 71 members (group vB). The average pre-test scores of these groups were 4.47 and 4.16 respectively. According to the independent sample t-test, groups vA and vB could also be considered as statistically equivalent ones ($p = 0.21 > 0.05$). The “Above average”/“Average”/“Promising”/“Others” distributions in groups vA and vB were 12/11/8/19 and 14/16/12/29 students respectively. (At these sub-group levels, also no significant differences were detected.)

We performed the experiment during the first week of the school year 2013–2014 (in computer labs where all students had individual access to the online e-learning tool). After they had received their group ID (A or B), students were asked to participate in the following three-phase e-learning session:

1. “Listen to It”: The first two slides presented a brief definition of what a sorting algorithm means. Accordingly, students from both groups were informed that the algorithm was going to be danced, animated, and user-orchestrated. They were made aware of the need to focus on the strategy and the ways that comparing and swapping operations are animated.
2. “Watch It”: Students from both groups were invited to watch (1) the folk dance illustration and (2) the computer animation of bubble sort algorithm.
3. “Try It”:
 - a. Students from group A had to orchestrate the algorithm on a random sequence stored in a white array.
 - b. Students from group B had to orchestrate the algorithm on a random sequence stored in a black array.

9.4.1 Results and discussion

The online environment registered the errors (and the type of errors) students had made, the amount of time they had spent with their task, and also counted their help requests. Based on the number of errors, group vB performed significantly better than group vA (see *Table 9.1*). At sub-group level, the means and the p-values were: “Above average” (5.21, 4.71: 0.43); “Average” (8.7, 3.2: 0.06); “Promising” (11.17, 4.16: 0.02); “Others” (12.65, 8.2: 0.08).

We focused on the following types of errors:

1. Although the next element pair to be dealt with was selected correctly, the operation ((C)omparison/(S)wap) to be applied on was chosen incorrectly: (a) S in place of C; (b) C in place of S;
2. Wrong element pair was selected when the next operation had to be: (a) C; (b) S.

(No significant differences were detected with respect to “Wrong parameter number”.)

Table 9.1. *Comparative analysis of students' performance scores (groups vA and vB)*

	Time/ step	Help	Errors	Wrong param- eter num- ber	Wrong action selected		Wrong param- eters selected	
					C	S	C	S
vA	5.05	1.58	9.6	1.48	1.42	0.62	5.32	0.76
vB	4.15	1.19	5.56	1	0.52	0.47	2.53	1.02
p	0.002	0.34	0.009	0.1	0.0002	0.24	0.006	0.28

Source: based on author's measurements

Type 1a errors indicated those moments when students performed directly the swapping operation without previously comparing the element pair in case. Whenever they omitted to compare initially right-ordered element pairs, the software registered type 2a errors. Comparing groups vA and vB with regard to these four types of errors, we found that group vA had made significantly more type 1a and 2a errors than group vB. No significant differences were detected between groups with respect to type 1b and 2b errors.

As mentioned above, the primary goal of white-/black-box tasks was to refine students' understanding of the studied algorithm. According to the constructivist view, the immediate feedbacks students received (with respect to the errors they had made) could result in progressively reconstructed mental pictures of the algorithm. To detect initial misconceptions, we have proposed to investigate how many students:

- skipped the explicit comparing operation of the first element pair that had to be swapped (first possible type 1a error),
- omitted to deal with the first element pair that was in the right order (first possible type 2a error).

Significant differences were detected between the groups. 30.77% of students belonging to group vA failed to avoid the first possible type 1a or 2a error. This percentage in the case of group vB was only 14.08%.

Interestingly, students belonging to group vA spent significantly more time with their tasks than their colleagues from group vB (see *Table 9.1*). A possible explanation: since group vB was exposed to less distracting factors, they could stay focused on the sorting strategy.

It can also be noticed that students from both groups (especially weakly performing ones) failed to profit enough from the presence of the help button (see *Table 9.1* – compare columns “Number of help requests” and “Number of errors”). For example, four students who made more than 50 errors had zero help requests. They preferred guessing instead of pushing the help button. (No significant differences were detected between groups with regard to the number of help requests.)

According to the above-presented results, as we expected, students working on white arrays had significantly more problems with comparing operations than those who had to sort invisible sequences. More specifically, students sorting visible sequences skipped significantly more frequently comparing operations than those working on a hidden sequence. A plausible explanation could be that students who saw the numbers performed the comparing operation in their mind, implicitly. Since students sorting black arrays did not see the numbers, they were forced, in the same sense as blind computers are, to explicitly perform the comparing operations in order to find out if the corresponding element pair had to or did not have to be swapped.

From the perspective of Epistemic Fidelity theory, these results can be interpreted as follows:

- In computers, AV invisibility could contribute to higher epistemic fidelity and, consequently, more effective learning.
- Wisely applied hiding results in higher epistemic fidelity, especially when learners are substituting computers in their algorithm processor role.
- High epistemic fidelity AV systems support not blind learners in identifying with blind computers.

We also observed that students working on black arrays tended to skip swapping operations more frequently than those who had to sort white arrays (see *Table 9.1*). A possible reason could be that since numbers were invisible they did not realize that comparison did not imply swapping either (although only comparison was animated). It seems that black-box tasks required more “implementation focus”. In addition, “black-box students” who had completely failed to grasp the logic of the algorithm during “Watch It” phases apparently had less chance to do this during the “Try It” phase. Among those who abandoned their orchestrating task, 60% had faced a black-box task. Among those who completed their task but made more than 50 errors (roughly equal to the number of steps the correct operation sequence had), 66% belonged to the “black-box group”. Black-box tasks were especially useful for that majority who previously grasped the logic of the algorithm but needed assistance in assimilating it from the perspective of a blind computer.

9.5 Conclusions

Teaching and learning computer algorithms is a challenging educational task. In order to elaborate the algorithm, students have to adjust to the primitive level of the computer. Thinking only of those primitive operations that computers can execute, they need to design the operation sequence that, as executed, results in the solution to the problem to be solved. Interactive AV has become a common method to help students to assimilate computer algorithms.

Effective AVs can be powerful supplementary, complementary, and alternative tools to written presentations or verbal descriptions (Shaffer et al., 2007). On the other hand, as we have discussed above, AVs might have their own specific weak points (compared to written/verbal descriptions): too much visualized information can be harmful.

By applying sequenced multiple AVs, the AlgoRhythmic environment generates a three-phase learning experience: (1) watching the dance choreography illustration, (2) watching the abstract computer animation, (3) participating in the white-/black-box algorithm orchestration. Phase 3 harmonizes with the constructivist approach to learning: learners become active participants of the AV experience. From the perspective of Epistemic Fidelity theory:

- During the first two phases, we tried to increase epistemic fidelity by common methods:
 - Adding graphical elements to the video recordings.
 - Dancers/elements that had reached their final positions “turned back”/“were recoloured”.
 - A pair of arrows directs user’s attention to the elements the current operation is applied to.
 - Expressive animations for comparison/swapping operations (The way we animate comparison operation transmits the idea of weighing the number pair to be compared by a balance).
- During black-box tasks, we proposed to increase epistemic fidelity in a new way: by applying invisibility.

While more research is needed to draw general conclusions in the studied topic (it is a limitation of this study that we investigated the proposed research questions only with respect to one specific sorting algorithm), the study we have performed reveals latent deficiencies that AV systems might have. Visualizing information that has extra meanings for human viewers can obstruct them in following strict computer algorithms. Research results show that wisely applied hiding may result in more effective AV due to its higher epistemic fidelity. As a final conclusion: Effective AV systems support not blind learners in assimilating the algorithm processing role of blind computers.

10 PROMOTING ALGORITHMIC/ COMPUTATIONAL THINKING OF BOTH SCIENCES- AND HUMANITIES-ORIENTED LEARNERS (STUDIES 6 AND 7)

Algorithmic and computational thinking (AT/CT) are important abilities in an information-based society, ones that all should possess. AT is related to the concept of creating and processing algorithms (Futschek, 2006). Since the term “algorithm” essentially refers to a sequence of logical steps aimed at performing a well-defined task, the creation of algorithms is mainly a human activity. Algorithms are everywhere in modern society. Many fields of modern life involve the processes of following procedures, applying protocols, or implementing techniques, all of which can be viewed as human-processed algorithms. Thus, a developed AT may be beneficial for a wide range of human activities. On the other hand, one of the main characteristics of the digital era is that the control behind the technology that has pervaded all sectors of society is implemented through computer-processed algorithms. Most people come into daily contact with computer-processed algorithms through information technology resources.

In 1999, the USA Committee on Information Technology Literacy of the National Research Council (NRC) called for an educational focus on Fluency with Information Technology (FITness). The NRC committee recommended a pedagogical approach that, besides contemporary skills and intellectual capabilities, incorporates ten fundamental IT concepts, including “algorithmic thinking and programming” and “modelling and abstraction”. In other words, being fluent in IT assumes an understanding of the basic concepts and principles of IT resources. From this perspective, AT is closely related to CT (Hu 2011). Accordingly, developing students’ AT should be included as an objective in all educational programmes at all levels and connected to lifelong learning. In addition, this educational issue implies that all students, including adult learners, should need to be familiar with what a computer algorithm is and how it works.

Hardly has any technology been as ubiquitous in human history as CS is today. The creation of a genuine educational programme which ensures a proper initiation into AT/CT is a major endeavour. As in most countries there is already a severe computational literacy gap between the different segments of society, this educational programme must be all-inclusive and must address all irrespective of age, gender, race, culture, orientation, or disability.

Accordingly, one of the most serious challenges faced by such an initiative is that of dealing effectively with diversity. In this chapter, we focus on one

specific facet of diversity: sciences- vs. humanities-oriented people. After Snow (1959) had introduced the term of “two cultures”, the concept of sciences-/humanities-oriented learner arose, and it was suggested that these two categories of people are characterized by different cognitive styles, often described as the individual difference in the way people acquire and process information (Witkin, Moore, Goodenough, & Cox, 1977; Achter, Lubinski, & Benbow, 1999; Billington, Baron-Cohen, & Wheelwright, 2007). On the other hand, taking learner diversity into account is a very complex task. Since motivation plays a crucial role in learning and constitutes a key element in all the approaches to active learning (Valle et al., 2011), the impact of diversity on learners’ motivation is a critically important aspect of the addressed educational issue.

The question we have proposed to analyse is as follows: Is it possible to create unified learning environments that promote AT/CT for all? This chapter includes two studies (Káta, 2015, 2020):

- 1) We present research results showing that properly calibrated learning environments have the potential to effectively promote the CT of both sciences-oriented and humanities-oriented students.
- 2) We analyse the specific motivational challenges that instructional designers could face in developing learning environments which bridge the diversity gap and target both learning communities.

10.1 The “two cultures”

Snow (1959) introduced the term “two cultures” in reference to sciences and humanities. Based on this concept, some authors speak about sciences-oriented and humanities-oriented learners according to their different ways of viewing the world and their different approaches to problem solving (Achter et al., 1999; Watters, 2010). The concept of cognitive style is often described as the individual difference in the way people acquire and process information (modes of perceiving, remembering, thinking, and problem solving) (Messick, 1976; Witkin et al., 1977; Ausburn & Ausburn, 1978). Additionally, Baron-Cohen, Knickmeyer, and Belmonte (2005) defined cognitive style as the interplay between two core psychological dimensions: empathizing and systemizing. Based on this concept, Billington and his colleagues investigated students of physical sciences and humanities (Billington et al., 2007). They found that the cognitive style was a much better predictor for students’ entry either into physical sciences or humanities than gender. These findings suggest that members of the “two cultures” are characterized by different cognitive styles.

The left/right brain learning theory supports this conclusion. According to this model, the two hemispheres of the brain control different modes of thinking. Experiments applying neuro-imaging technologies have shown that activities

involving logical, sequential, rational/analytical thinking and bottom-up processing are more closely associated with the left side of the brain (“academic brain”). Conversely, activities focusing on aesthetics, imagination, creativity, feelings, and top-down processing are more active in the right hemisphere (“artistic brain”). Accordingly, students enrolled in humanities programmes show a greater degree of right brain activity than those studying sciences (who use a greater degree of left brain activity) (Kagan, 2009; Pobric et al., 2008). So, perhaps not surprisingly, students with dominant left brain thinking have a preference for sciences, whilst right brain thinkers favour humanities subjects. Despite these facts, people cannot be categorized as exclusively left-brained or right-brained learners. Furthermore, recent research results emphasize that optimal learning implies a balanced involvement of both sides of the brain (Bransford et al., 1999; Eisenhower SCIMAST, 1997; Kátai, Juhász, & Adorjáni, 2008).

The above findings suggest that whereas AT/CT (as a cognitive competence that implies dominant left brain activity) is important for all students, it could be less familiar to humanities-oriented learners than to their sciences-oriented colleagues. The immediate response of modern pedagogy to diversity is differentiation. Since designing and implementing effective differentiated teaching-learning strategies may involve substantial additional costs, we proposed to investigate the following topics: Are differences (with respect to AT/CT) between humanities- and sciences-oriented students unbridgeable? Should they be approached separately or as carefully designed and properly calibrated (maybe adaptive) strategies that fit both learning communities?

Unfortunately, the literature lacks studies that address AT as a valuable competence for all students (i.e. regardless of the educational programme they are enrolled in). Research on this issue has predominantly focused on teaching-learning algorithms within the confines of computer programming classes or courses (Knuth, 1985; Milkova, 2005; Futschek, 2006; Futschek, 2007; Futschek & Moschitz, 2010; Douadi et al., 2012). Schwank (1993) studied AT in relation to the teaching of mathematics. In this chapter, we analyse AlgoRythmics environment from the following perspective: online self-paced e-learning tool to enhance all students’ AT and to introduce them to how computer algorithms work.

We proposed to move the groups closer to each other by adding effective didactical elements and not removing significant content. We focused on such algorithms that would be relevant (and accessible) for humanities-oriented students and important (not trivial) for their sciences-oriented colleagues as well. Regarding the implemented didactical strategy, we expected that it would prove to be critically important for humanities-oriented students and also enhancing for sciences-oriented ones. We designed the learning experience taking into account principles of constructivist learning theory, research results in algorithm visualization, and principles of motivational design.

10.2 Promoting algorithmic/computational thinking

The term CT was originally introduced by Wing (2006). In our studies, we use a revised definition of the term: the thought process involved in formulating problems so that “their solutions can be represented as computational steps and algorithms” (Aho, 2012). From this approach, it is again clear that CT and AT are strongly related concepts. Denning (2009) also emphasizes that algorithms are central to CT. More recently, he underlines that algorithms must control some computational model, and step sequences that require human judgment should not be considered algorithms in the context of CT (Denning, 2017). As detailed in *Chapter 7*, we have proposed to enhance CT by inviting students on a “delicious algorithmics tasting tour” (in the micro-world of sorting algorithms: bubble sort, insertion sort, selection sort, shell sort, quick sort, and merge sort) based on multiple algorithm visualizations. The learning environment we created facilitates the simulation-based study of the selected algorithms in a relevant, accessible, and engaging way (supporting users in assimilating the algorithm-processing role of computers). It can be set up to generate the following learning experiences (see *figures 7.1–7*):

- 1) the algorithm is visualized by a videotaped “sequence of folk dancers” wearing the numbers to be sorted on their dresses;
- 2) the algorithm is animated on a white-box array (the number sequence is visible) followed by:
- 3) student-reconstructed and
- 4) orchestrated animations;
- 5) students are invited to orchestrate the studied sorting algorithm on a black-box array (the number sequence is hidden; the user is informed only about the results of the comparison operations);
- 6) the algorithms are visualized as they work side by side on six different colour scale bars.

Of course, such a learning experience is only a first step in developing students’ CT. Definitions of CT emphasize that cultivating this skill implies more than assimilating ready-made procedures. It implies devising procedures. For example, Selby and Woollard (2013), after investigating several definitions, conclude that CT “is a focused approach to problem solving, incorporating thought processes that utilize abstraction, decomposition, algorithmic design, evaluation, and generalizations”. Studying basic computer algorithms in the AlgoRythmics environment could be a good starting point in this sense, especially if we choose to improve students’ CT by computing education (Guzdial, 2008; Tedre & Denning, 2016).

The constructivist learning theory provided the main theoretical framework for this research. Constructivist learning environments are student-centred, engaging, and reflective and make it possible for students to learn from their

experiences (Jonassen, Peck, & Wilson, 1999). According to Wang (2009), such a learning environment assumes attentive and thoughtful design. We purposefully focused on implementing the following previous research results regarding effective CT promoter learning environments.

- Why visually illustrated algorithms? Since computer algorithms are inherently abstract dynamic processes, AV has become the common approach to make them more tangible. We chose to visually illustrate how algorithms work by videotaped dance performances and computer-based animations (Shaffer et al., 2010). The meta-analysis performed by Höffler and Leutner (2007) emphasizes the educational superiority of representational animations compared to static pictures, especially when procedural-motor knowledge has to be assimilated.
- Why sequenced multiple representation? The basic idea of using sequenced multiple AVs is that users can benefit from the properties of each representation (Meij & Jong, 2006). Two key attributes of multiple representations are complementarity and redundancy. Redundancy is essential to make learners able to relate to different representations. Complementary attributes can be used to implement the principle of progression with respect to the informational content, complexity, level of abstractness, and the control the learner has in the algorithm animation process. We use four different representations of the number sequence to be sorted: embodied by a dancer sequence, stored in a white-box array, stored in a black-box array, and illustrated as a colour scale bar.
- Why interactive learning environment? The meta-study (Hundhausen et al., 2002) stresses the decisive role interactivity has in effective AVs. This study concludes that AVs promote effective learning when users are engaged actively in the visualization process (instead of passively viewing it). In other contexts, the same phenomenon was observed (e.g. Mork, 2011). To implement this principle of “genuine active involvement”, the software we designed invites users to orchestrate the studied algorithms (Mayer & Chandler, 2001).
- Why applying selective hiding? As detailed in *Chapter 9*, applying hiding may support human viewers in assimilating the algorithm processing role of blind computers due to their higher epistemic fidelity (Kátai, 2014a). During the black-box-based algorithm orchestration processes, since the stored numbers are invisible, users are forced (as computers too) to perform the comparison operations explicitly (not only implicitly, in their minds) in order to realize whether the corresponding elements need or need not to be swapped.
- Why pattern-recognition-oriented strategy? Since algorithms are in fact generalized patterns intended to solve problems, CT assumes pattern recognition and generalization skills (Wu & Richards, 2011). Humans are

unbeatable pattern recognizers in most instances (Jain, Duin, & Mao, 2000). Obviously, learning from examples assumes carefully selected examples (Jain & Duin, 2004). The interactive learning environment we created implements the following pattern-recognition-oriented CT promoter method: (1) students are presented with illustrations of the algorithm on carefully selected sample inputs and (2) students are invited to enhance and refine their understanding by orchestrating the algorithm on random inputs.

10.3 The motivational perspective

The impact of diversity on education has been researched from several perspectives. One of them is the motivational perspective. On the other hand, in motivational research, little attention has been paid to studying the impact of orientation (sciences/humanities) on motivation. Additionally, the specific motivational challenges of developing students' CT have also been poorly researched. In this chapter, we focus on the instructional and motivational design aspects of the AlgoRhythmic learning environment. We tried to identify motivational principles that could have key roles in arousing and sustaining students' (science-/humanities-oriented) motivation during e-learning processes that aim to promote CT.

Motivation helps people to pursue goal-directed behaviours and activities such as learning. It serves to energize students, providing intensity and direction. Motivation theorists distinguish between intrinsic motivation (referring to the internal drive; doing something because it is inherently interesting or enjoyable) and extrinsic motivation (arising from factors outside the individual; doing something because it leads to a separable outcome) (Deci & Ryan, 1985). Although developing successful motivational strategies assumes an optimal combination of these two types of motivational resource (Omrod, 2002), the outstanding role that intrinsic motivation has in promoting high-quality learning is unquestionable (Fair & Silvestri, 1992; Martens, Gulikers, & Bastiaens, 2004). Accordingly, motivational design is a systematic process to make instruction more intrinsically interesting (Keller, 1983).

According to motivation theorists, learning environments that engage students in the learning process yield stronger intrinsic motivation (Lepper, Henderlong, & Iyengar, 2005; Robertson & Howells, 2008). Research in this field has revealed some major factors supporting students' motivation. Three of them are positive emotions, moderate-progressive challenge, and active involvement, which may have a distinguished role in computer-mediated environments (Finneran & Zhang, 2005). Research results in game-based learning also inspired us in selecting these motivational behaviours. For example, Hamari et al. (2016) analysed the impact of challenge, engagement, and immersion on learning in

game-based learning environments. The sense of immersion is directly related to the emotional composition of the learning experience (Fassbender, Richards, Bilgin, Thompson, & Heiden, 2012). Studies on serious games also emphasize the importance of players' emotion by describing it as the main player characteristic considered to be important for learning processes and performance (Schrader, Brich, Frommel, Riemer, & Rogers, 2017).

Recent evidence show that emotions are not only outcomes of motivated behaviour, but they also influence the cognitive processes associated with motivation (Triberti, Chirico, La Rocca, & Riva, 2017; Um, Song, & Plass, 2007). For example, emotion often underlies curiosity, which is commonly regarded as a prime promoter of intrinsic motivation (Malone & Lepper, 1987; Litman, 2005). Research on challenge as a motivator shows that promoting challenge–skills balance is optimally motivating (Nakamura & Csíkszentmihályi, 2002; Turner & Meyer, 2004; Ott & Tavela, 2010). Relevant active involvement could be decisive in promoting intrinsic motivation since it may have a crucial influence on sustaining students' engagement during the learning process until the knowledge construction has been completed (Lepper & Malone, 1987; Garris, Ahlers, & Driskell, 2002).

Inviting learners to participate in sequenced learning units generates additional motivational challenges. The output emotions of the current phase of a learning session are input emotions for the next phase (Wlodkowski, 1985). A relevant active role in a challenging learning session contributes to a sense of achievement. In addition, the way students think about the next learning unit generates corresponding emotions. Perceiving inappropriate challenges may result in anxiety or boredom (Csíkszentmihályi, 1990). On the other hand, the probability of success can promote exciting expectations. Teachers may have a key role in promoting the above “learning ingredients” (Christophel, 1990), but technology has also its own strengths (Barger & Byrd, 2011).

The study (Rovai, Ponton, Wighting, & Baker, 2007) concludes that online e-learning may foster stronger intrinsic motivation than traditional classroom learning. On the other hand, Ott and Tavela (2010) emphasize that “new millennium learners” (Pedró, 2006) are not indifferent to the quality of the e-learning experience. Most frequently recalled deficiencies of e-learning materials are: lack of edge and emotion, absence of a great beginning and ending, too much sameness, etc. On the other hand, if e-learning materials are interesting only – for example – because they are novel, then they may lose their appeal as learners become accustomed to them. It is clear that simply adding some multimedia elements is not enough to significantly increase and sustain intrinsic motivation (Hamid, 2001; Clark & Mayer, 2002; Keller & Suzuki, 2004; Martens et al., 2004). Therefore, we have proposed to implement the following motivational strategy:

- Arousing motivation: During the “dance performance phase”, the focus is on arousing curiosity by combining science with art and the modern with

the traditional. In order to test if the created “algorithmic dances” have the potential to provide novelty, incongruity, and surprise (Keller, 1983; Berlyne, 1960), we posted them on the YouTube website. Users’ reactions confirmed our expectations (for details, see *Chapter 7*). Artistic elements play a key role in attracting humanities-oriented people. The presence of culture may also promote emotional motivation.

- Sustaining motivation: The role of the “animation phase” is to help students to focus on the key elements of the algorithm (Mayer, 2003; Kalyuga, Chandler, & Sweller, 1999) and to prepare them for the “doing phases”. During the “doing phases” (reconstruction, “white-box orchestration”, “black-box orchestration”) of the e-learning session, students are invited to actively participate in the animations (Mayer & Chandler, 2001) according to the principle of moderate-progressive challenge. According to constructivist learning theory, learning from their experiences and applying the knowledge they have just gained may result in more effective learning (Jonassen, Peck, & Wilson, 1999).
- In the conclusion part, the “sorting movie” that we created (displaying the parallel sorting of the six colour scales) aims to arouse aesthetic emotions and to provide a global view of the studied topic. Aesthetic emotions can contribute to a great ending to the learning experience (Parrish, 2009; Riaz, Rambli, Salleh, & Mushtaq, 2011).

As we mentioned above, two other important motivational factors related to the moderate challenge are anxiety (fear of failure) and probability of success. In line with Keller’s ARCS (attention, relevance, confidence, and satisfaction) model of motivation (Keller, 1987), in order to guarantee that students can accomplish the tasks (and to reduce anxiety), we provided them with help buttons that indicate the next operation to be performed. Moreover, each animation task can be repeated (for different randomly generated inputs) until it is solved correctly and without using the help buttons. This option aims to promote the desired sense of achievement.

10.4 The experiment

With respect to the requirement of “carefully selected examples”, we have chosen sorting algorithms because these are probably the most directly perceived computer algorithms by IT users. Sorting a list is a common operation in many fields of work and is one of the most fundamental problems in CS. As mentioned above, the AlgoRythmics learning environment has been designed to introduce students to the micro-world of comparison-based sorting algorithms. Bubble sorting (“optimized” version) was selected for this experiment, too, since it applies a relatively easily detectable algorithmic pattern (it compares and swaps

neighbouring elements and traverses the list, again and again, in quite a similar way). The input sequences for the illustrations of the algorithm were also selected carefully (for example, to emphasize that a single traverse of the sequence can result in placing more than one element to its final position).

The investigation involved 48 undergraduate students from Sapientia Hungarian University of Transylvania. The independent variable was students' orientation (25 (S)cience-students, 23 (H)umanities-students). The percentages of females were 12% (S-students) and 82% (H-students), resp., which is characteristic in the case of these educational programmes. H-students were IT users but strangers to computer programming (87% of them had not attended any programming-oriented courses during their high-school education; the rest had only some minimal CS concepts included in their high school curriculum). We performed the experiment after the S-students had been initiated into programming. Both groups studied the selected sorting algorithm by following the above-presented six-stage learning session. To be able to measure students' performance results, the software registered all the errors they made and their help requests.

We designed the experiment so that we could also measure the motivational impact of the generated e-learning experience. Although there are many motivation questionnaires used in educational psychology, we have proposed the development of a specific questionnaire sequence for this study. Several studies (Blumenfeld, 1992; Blumenfeld & Meece, 1988; Lee & Anderson, 1993; Lee & Brophy, 1996; Weiner, 1990) conclude that students may express different motivational traits when studying specific subject content areas. According to the classic model of motivational psychology, personal and situational factors influence the current motivation, which in turn influences learning. Rheinberg, Vollmeyer, and Burns (2001) proposed a questionnaire with 18 items for assessing the current motivation of students working on a specific task. Researchers such as Vollmeyer, Burns, and Rheinberg (2000) and Friedl et al. (2006) used this instrument to investigate the motivational characteristics of students learning in interactive multimedia environments. We adapted this method and worked out a set of nine specific questionnaires (Q1–9; see the Appendix) to detect the level and type of students' motivation during the e-learning experience we designed.

The questionnaire items aimed to detect the way students think and what they are feeling before and after the e-learning session (unit) and between the consecutive stages (sub-units). While the questionnaire proposed by Rheinberg, Vollmeyer, and Burns (2001) is built around factors such as probability of success, anxiety, interest, and challenge, we have focused on assessing the motivational contributions of the generated (situational factors) positive/negative emotions, the challenge and active involvement during the e-learning experience. We expected to find significant differences between H- and S-students.

Each questionnaire item included a statement and a response scale. We used a 7-point scale (1: Strongly Disagree ... 7: Strongly Agree). We coded the items

as follows: Letters E, C, and I refer to Emotions, Challenge, and Involvement. “E-items” aim to detect emotions resulting from three sources:

- How interesting the previous e-learning stage was and the next one seems to be.
- How challenging the previous e-learning stage was.
- How “exciting” student’s involvement in the previous e-learning stage was.

“C-items” aim to detect the level of challenge the student anticipates in the next e-learning sub-unit. The “I-items” aim to detect the way the students think and what they are feeling regarding their forecasted involvement in the next e-learning sub-unit.

The experiment was conducted in computer labs, where all students had individual access to the online e-learning tool. Before the e-learning session started, students had been briefly initiated into or reminded of concepts such as: Why CT is important? What is an algorithm? Why could studying sorting algorithms be beneficial? What is a sorting algorithm? In addition, a brief overview of the six-phase e-learning session was presented to both groups. Following the above-presented introduction (hearing phase), students were invited:

- (Q1) to answer 5 questions regarding algorithms and CT;
- (Q2) to answer 11 questions regarding the anticipated e-learning experience;
- s1: (seeing phase 1: arousing/stimulating motivation) to watch the danced algorithm; (surprising science–art, modern–traditional combinations) (*figures 7.1 and 7.7.a*);
- (Q3) to answer 9 questions regarding the “dance stage” that had just finished and the “animation stage” that was about to start;
- s2: (seeing phase 2: supporting abstracting) to watch the animation of the algorithm (*Figure 7.7.b*);
- (Q4) to answer 12 questions regarding the “animation stage” that had just finished and the “reconstruction stage” that was about to start;
- s3: (doing phase 1: active involvement, progressive challenge) to reconstruct the operation (compare, swap) sequence of the observed animation (*Figure 7.7.c*);
- (Q5) to answer 17 questions regarding the “reconstruction stage” that had just finished and the “orchestration stage” that was about to start;
- s4: (doing phase 2: active involvement, progressive challenge) to orchestrate the algorithm on a randomly generated sequence stored in a white-box array (the numbers being visible) (*Figure 7.7.d*);
- (Q6) to answer 17 questions regarding the “orchestration stage” (“white-box stage”) that had just finished and the “black-box stage” that was about to start;
- s5: (doing phase 3: active involvement, progressive challenge) to orchestrate the algorithm on a randomly generated sequence stored in a black-box

- array (being informed about the results of the comparison operations) (*Figure 7.7.e*);
- (Q7) to answer 13 questions regarding the “black-box stage” that had just finished and the “parallel simulation stage” that was about to start;
- s6: (arousing aesthetic emotions) to watch a nice parallel simulation of the six sorting algorithms (different sorting algorithms are visualized as they are working side by side on different colour scale bars) (*Figure 7.7.f*);
- (Q8) to answer 4 questions regarding the “parallel simulation stage” that had just finished;
- (Q9) to answer 9 questions regarding their global impressions on the e-learning session they had just finished.

10.5 Results regarding students' performance

As already mentioned, the software registered all the errors students had made (including the type of errors: wrong parameters or wrong action) and the number of requests for help. (Students were made aware of this beforehand.)

Our first analysis (*Study 6*) focused on the following research questions:

- What kind of differences can be observed between the performance results of sciences- and humanities-oriented students in an AT/CT promoter learning environment?
- Can differences be bridged by properly calibrated (possibly adaptive) learning strategies?

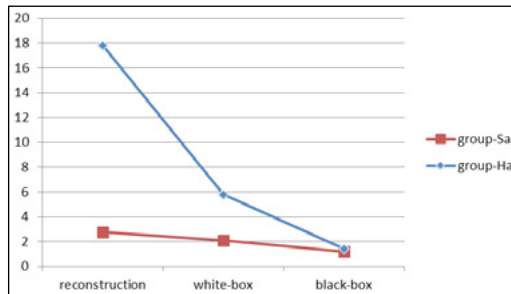
The dependent variable was the level the students attained in the studied algorithm. We selected as testing points: phase 3 (reconstruction task), phase 4 (white-box task), and phase 5 (black-box task). Students' orientation (S-students (science); H-students (Humanities)) was the independent variable, and the type of learning experience was a controlled variable. We expected that:

- S-students would assimilate the algorithm faster than their H-colleagues.
- The majority of H-students would also understand the algorithm at an acceptable level.
- Carefully designed and properly calibrated learning strategies could fit both groups, moving them closer to each other.

The evaluation process was based on the number of errors made by the students as well as on the number of help requests they made. Analysing the on-line reports, we concluded that 95.65% of the S-students (group-Sa) assimilated the studied algorithm. In the case of H-students, this percentage was 73.91% (group-Ha). These conclusions reflect the students' performance during their last task (orchestrating the algorithm on a random sequence stored in a black-box array). Focusing on those students who assimilated the algorithm, the average of the errors they made decreased as follows:

- Group-Ha: (1) 17.82 (during the reconstruction task); (2) 5.82 (during the white-box orchestration); (3) 1.41 (during the black-box orchestration).
- Group-Sa: (1) 2.77 (during the reconstruction task); (2) 2.09 (during the white-box orchestration); (3) 1.22 (during the black-box orchestration).

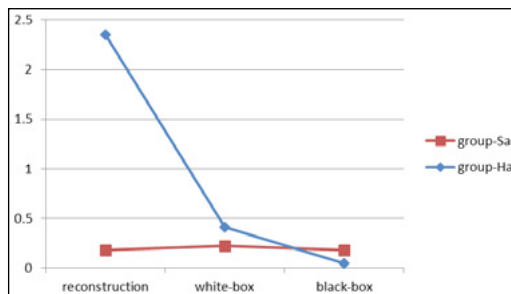
Interestingly, the differences between the corresponding numbers of errors decreased as follows (we applied independent sample t-tests): (1) 15.05 (large, $p = 0.0001 < 0.05$); (2) 3.73 (significant, $p = 0.0427 < 0.05$); (3) 0.18 (“equality”, $p = 0.3499 > 0.05$) (see *Figure 10.1*).



Source: Kátai, 2015

Figure 10.1. Number of errors students made during the reconstruction, white-box and black-box phases

Regarding the number of help requests made, we observed a similar phenomenon. During the reconstruction task, H-students had to access the help button significantly more frequently than their S-colleagues (group-Ha: 2.35; group-Sa: 0.18; $p = 0.0012 < 0.05$). During the white-box and black-box phases, these large differences between groups disappeared almost entirely (see *Figure 10.2*).

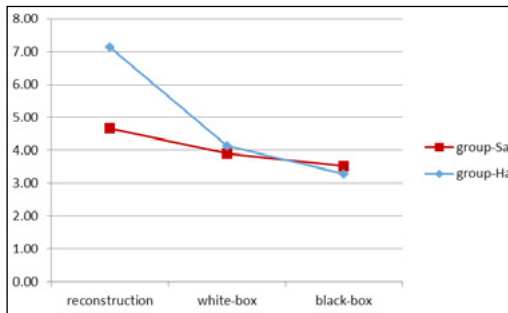


Source: Kátai, 2015

Figure 10.2. Number of help requests students made during the reconstruction, white-box and black-box phases

We also analysed students' time management during the e-learning session. We computed the average amount of time they had spent on their actions (wrong answer, help request, good answer). During the reconstruction task, H-students spent significantly more time on their actions than S-students did (group-Ha: 7.14 sec/action; group-Sa: 4.68 sec/action; $p = 0.0005 < 0.05$). During the white-box and black-box phases, the differences between the groups became negligible (see *Figure 10.3*):

- white box (group-Ha: 4.14 sec/action; group-Sa: 3.91 sec/action; $p = 0.23 > 0.05$);
- black box (group-Ha: 3.28 sec/action; group-Sa: 3.53 sec/action; $p = 0.11 > 0.05$).



Source: Kátai, 2015

Figure 10.3. *The amount of time students spent on their actions (wrong answer, help request, good answer) during the reconstruction, white-box and black-box phases*

Accordingly, although H-students' performance results were superior to those of their S-colleagues (as we expected), there was a strong correlation (number of errors: $r = 0.94 > 0.87$) between them, and differences diminished as both groups advanced with their e-learning tasks. The large initial differences or inconsistent responses can partially be explained by the unfamiliarity of H-students with the studied topic and the nature of the required interactivity (Kaminski et al., 2009).

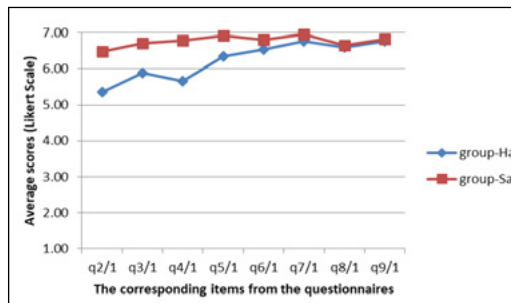
10.6 Results regarding the motivational perspective

Regarding the motivational perspective (second analysis, *Study 7*), we addressed the following research questions:

- What motivational principles could play a key role in arousing and sustaining students' motivation during e-learning processes that aim to promote AT/CT?

- Are motivational differences (with respect to AT/CT) between humanities- and sciences-oriented students unbridgeable?
 - Can AT/CT promoter learning environments (supporting students to assimilate and process computer algorithms) be calibrated in such a way to be motivating for all students?

A reliability analysis was carried out on the items of the motivation questionnaires. Cronbach's alpha, the measure of internal consistency for Likert-type scales, showed that the questionnaire items as grouped in the analysis reached acceptable reliability (for *Figure 10.4*, $\alpha = 0.92$; for *Figure 10.5*, $\alpha = 0.84$; for *Figure 10.6*, $\alpha = 0.88$; for *Figure 10.7*, $\alpha = 0.79$; for *Figure 10.8*, $\alpha = 0.87$). In all cases, all items included in the analysis appeared to be worthy of retention as the deletion of none of them would have resulted in an increase of the internal consistency of the measure. (In the following, notation qx/y refers to item y of questionnaire x; notation qa-b/y refers to item y of questionnaires a..b; notation qx/a-b refers to items a..b of questionnaire x.)



Source: Kátai, 2020

Figure 10.4. *Appreciation scores*

Generally speaking, both groups positively appreciated the e-learning experience. *Figure 10.4* shows groups' appreciation scores during the learning process: (q2/1) "I liked the idea behind this e-learning session"; (q3-8/1) "I liked this phase of the e-learning session"; (q9/1) "I liked this e-learning session". Groups' scores referring to the initial statement (q2/1) were 5.35 points (group-Ha, 72.5%) and 6.46 points (group-Sa, 91.33%). We managed to sustain and even gradually increase these initial levels of motivation. Groups' scores regarding the global impression statement (q9/1) were 6.76 points (group-Ha, 96%) and 6.83 points (group-Sa, 97.17%). To test if differences between groups (regarding the whole learning process) were significant, we performed a MANOVA. The grouping variable was students' orientation. We chose as dependent variables students' scores referring to statements q2–9/1. Test results confirmed our expectation that S-students mostly appreciate this kind of learning experience ($p = 0.0009$).

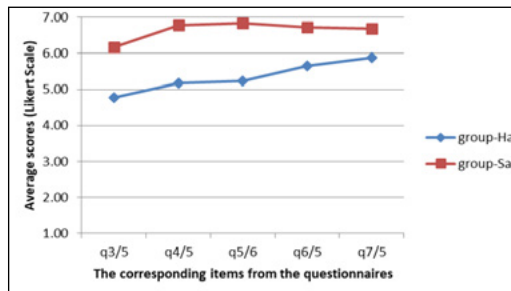
Comparing (with paired sample t-test) the means of H-students' scores with respect to statements q2–4/1 (“hearing and seeing phases”: 5.63 points, 77.17%), with the means referring to statements q5–7/1 (“doing phases”: 6.55 points, 92.5%), we detected a significant increase ($p = .0000$) in their appreciation level. This increase significantly diminished the size of the differences between groups. Interestingly, this increase took place in spite of the fact that students made quite a lot of errors during the “reconstruction task” (*Figure 10.1*).

Regarding other global impression statements such as “I liked this e-learning session because of its interestingness/challenge/interactivity” (q9/2–4), S-students' appreciations were superior to those of H-students', but no significant differences were detected. Additionally, both groups set up the following relations between the motivational contributions of the studied factors: “interestingness” < “challenge” (group-Ha, $p = 0.0133$; group-Sa, $p = 0.03$) and “challenge” < “interactivity” (group-Ha, $p = 0.2313$; group-Sa, $p = 0.003$) (we applied paired sample t-tests). Since the dance choreography illustration was the most distinctive part of the generated learning experience, we compared the two groups' appreciation scores regarding this component before (q2/2: “I consider that combining arts with sciences is an interesting idea.”) and after (q9/5: “I liked this e-learning session because of the dances.”) the learning session. While H-students reported approximately the same appreciation level at the end as at the beginning of the experiment (83% vs. 77%), S-students' appreciation increased significantly (87% vs. 97%; paired sample t-test, $p = 0.04$). In other words, while both groups indicated equally high scores in anticipation (group-Ha: 82%, group-Sa: 87%), the appreciation score reported after the learning session by group-Sa was significantly higher than group-Ha's score (group-Ha: 77%, group-Sa: 97%; two sample t-test, $p = 0.0002$).

Comparing (MANOVA) groups' scores referring to the statements “I grasped the logic of the sorting strategy” (q3/5, q4/5) and “I reached this performance because I understood the strategy the algorithm applies” (q5/6, q6/5, q7/5), we found significant differences ($p = 0.0014$) in the favour of S-students (*Figure 10.5*). Examining the graphics separately, we noticed the following:

- Group-Sa reported sharp increases in their understanding after they had watched the dance performance and the animation. Performance results more or less confirmed these estimates (*Figure 10.1*). During the “doing phases”, they adjusted their estimates to correspond better with their real level of understanding.
- In the case of group-Ha, a gradual increase can be observed in students' self-estimated levels of understanding. However, performance results did not confirm these scores (*Figure 10.1*). During the reconstruction task, H-students realized that they had not understood the algorithm adequately, and after this phase they did not report further increases in their level of understanding. Group-Ha's estimates were (relatively) consistent with

their performance results only after the white- and black-box tasks. Thanks to the “doing phases” of the learning process, H-students’ sensations that they grasped the logic of the strategy were followed by real increases in their understanding.

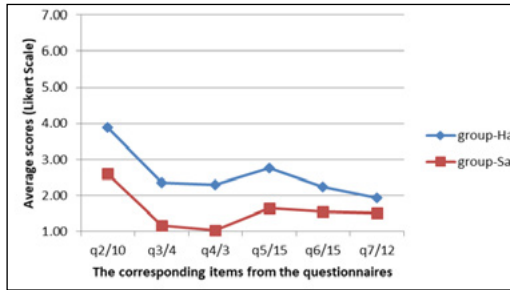


Source: Kátai, 2020

Figure 10.5. *Estimated understanding*

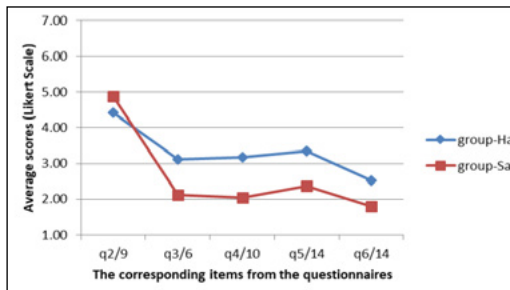
Figure 10.6 shows the groups’ scores referring to the statements “I am afraid/consider that what this e-learning session implies is beyond me.” (q2/10, q5/15, q6/15, q7/12) and “I did not realize what was going on.” (q3/4, q4/3). Both groups reported relatively low-level feelings of anxiety, and a strong correlation can be observed between their responses ($r = 0.86 > 0.70$). Although H-students’ scores were significantly superior to those of their S-colleagues (MANOVA, $p = 0.0056$), the size of the differences diminished as the two groups advanced during the learning process. These graphics also reveal that both groups overestimated their level of understanding after the “seeing phases” and emphasize the outstanding importance of the “doing phases” in such learning topics. We also observed strong correlations between students’ responses to the statements (q5/15, q6/15, q7/12) and the number of errors they made during the “doing phases” (group-Ha, $r = 0.99 > 0.87$; group-Sa, $r = 0.98 > 0.87$) (see Figure 10.1).

After students had watched the danced algorithms, both groups understood more clearly what a sorting strategy means. The moderate challenge students perceived starting from this point (Figure 10.7; q3/6: “This e-learning session seems to be a challenging one.”; q4/10: “Reconstructing the same operation sequence does not seem to be a complicated task.”; q5/14: “Orchestrating this strategy on random input sequences will be a challenging task.”; q6/14: “Since I grasped the logic of the strategy, it is not a problem for me to orchestrate it on hidden sequences.”) explains their high global impression scores regarding the motivational contribution of the challenge factor (q9/3). Although the challenge perceived by group-Ha was superior to group-Sa, the difference between them was not significant (MANOVA, $p = 0.2$).



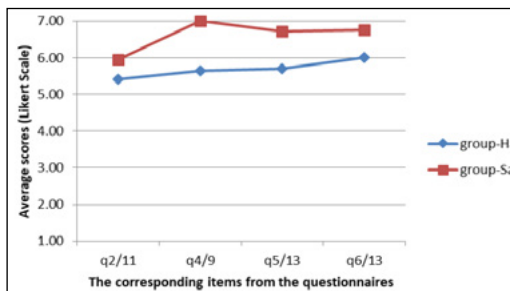
Source: Kátai, 2020

Figure 10.6. *Level of anxiety*



Source: Kátai, 2020

Figure 10.7. *Perceived challenge*



Source: Kátai, 2020

Figure 10.8. *Active involvement*

On the other hand, there was a strong correlation ($r = 0.94 > 0.75$) between groups' responses, and the differences diminished towards the end of the learning session.

Figure 10.8 shows the scores reported by the two groups on the subject of students' forecasted involvement in following the learning unit (q2/11, q4/9, q5/13, q6/13; "I appreciate that the software makes possible to do (reconstruct/orchestrate) ..."). The differences are significant (MANOVA, $p = 0.0008$). After the "seeing phases", both groups were more motivated to be involved in the learning process than at the beginning of the session (group-Sa reported maximal score at this point). Considering the whole learning process, H-students' scores indicate a continuous increase.

10.6.1 Limitations

A first limitation of these studies is that the majority of H-students were females and most S-students were males (as mentioned above, this distribution is characteristic of these educational programmes). This fact could affect our results. For example, the significant differences we detected between groups regarding the anxiety factor could partly be caused by gender differences. This would be in line with several studies that investigated the effect of gender on computer anxiety. For example, McIlroy et al. (2001) report persisting gender differences in self-reported computing anxiety. In a similar study (performed in the same country where our investigation was implemented), the authors also revealed a significant gender effect with respect to computer anxiety (Durdell & Haag, 2002).

The only component where H-students' scores were higher than those of their S-colleagues is participants' appreciation regarding their forecasted involvement in following the learning unit (see *Figure 10.8*). But this result could also be perturbed by gender differences. For example, Khan, Ahmad, and Malik (2017) report that in the game-based learning context they have analysed girls outperformed boys in terms of engagement.

Another limitation of our approach is that the learning session we designed included only one algorithm, a specific sorting algorithm. In addition, definitions of AT/CT emphasize that promoting this skill involves more than supporting students in assimilating basic computer algorithms (Shute, Sun, & Asbell-Clarke, 2017).

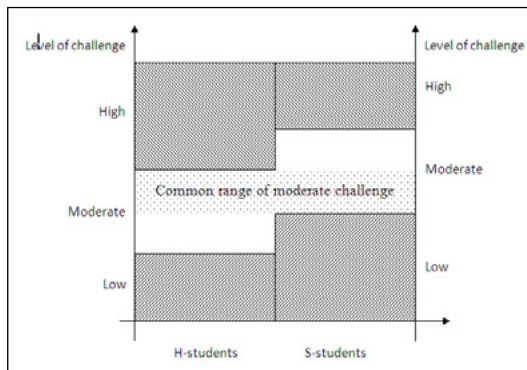
10.7 Conclusions

One of the main conclusions of this research is that there are no unbridgeable differences between the ways H- and S-students relate to AT/CT promoter e-learning tools. We found strong correlations between both the performance

results and motivational scores of the groups. Additionally, S-students' superiority over their H-colleagues (in both aspects) progressively and significantly diminished as the groups advanced with their e-learning tasks.

Findings of the first analyses stress the outstanding role of meaningful active involvement in promoting AT/CT. Whilst the majority of S-students recognized the logic of the sorting strategy during the "seeing phases", H-students understood it mostly by orchestrating it. S-students also refined their understanding during "doing phases".

Because the principle of moderate challenge covers a relatively large range of challenge, we were able to calibrate the e-learning tool so that it was moderately challenging to both the S-students and their H-colleagues (common range of moderate challenge; see *Figure 10.9*). After we posted the "sorting dances" on YouTube's AlgoRhythmic channel, the optimization elements of the algorithm generated repeated debates among CS students too. In conclusion, the above-presented research results attest that substantial CS content can be made digestible even for humanities-oriented learners by carefully designed content presentation. In this sense, principles of progressively challenging tasks and gradual shift from concrete to abstract could have key roles. Adapting e-learning sessions to users' needs (by skipping or repeating certain units) could also be a good method for keeping the challenge factor at optimal level. Whilst H-students needed all three "doing phases", the "reconstruction phase" could be skipped in the case of S-students.



Source: Kátai, 2020

Figure 10.9. *The way the level of challenge may depend on students' orientation (in algorithmic thinking promoter learning environments)*

Adaptive Resonance Theory also provides explanations why the presented "hearing, seeing, doing" learning strategy might be an effective pattern-recognition-oriented AT/CT promoter method. According to Grossberg (2013), excitatory

matching and attentional focusing on bottom-up data using top-down expectations generates resonant brain states leading to a resonant state that focuses attention on a combination of features (the “critical feature pattern”) that is needed to correctly classify the input pattern at the next processing level and beyond. The “hearing phase” generates expectations for the “seeing phases” and these ones for the “doing phases”.

Research results of the second analysis provide new insights into the motivational behaviour of sciences- and humanities-oriented learners within an AT/CT promoter learning environment. S-students’ superiority with regard to the majority of test items is a quite evident result and is in line with previous studies in the topic of cognitive style characterized by the psychological dimensions of empathizing and systemizing (Baron-Cohen, 2003). Billington, Baron-Cohen, and Wheelwright (2007) studied the cognitive style of sciences- and humanities-oriented students. Their research evidence indicates a “systemizing > empathizing” profile for sciences students (as a group) and an “empathizing > systemizing” profile for the humanities ones (as a group). Obviously, studying algorithms implies, first of all, systemizing, which is defined by the authors of the aforementioned study as the drive and ability to analyse a system in terms of an INPUT – OPERATION – OUTPUT principle. The surprising finding of this research is the way that H-students have caught up with their S-colleagues.

Sequenced multiple representation, besides its direct cognitive benefit (implementing the principle of gradual shift from concrete to abstract), allowed us to explore multiple motivational resources. Whilst students considered the dance performance mainly to be interesting, they found the black-box task to be exciting (because of its challenge and interactive character). Whilst the dance performance helped students to imagine what a sorting algorithm means, the black-box representation helped them to focus on the key elements of the strategy (on which sequence of element pairs, what operations [compare or compare + swap] should be performed). For example, Kátai (2014a) reported that students sorting hidden sequences skipped significantly less frequently comparing operations than those working on visible sequence (for details, see *Chapter 9*).

Another conclusion of this research is that – in AT/CT promoter environments – the principles of “moderate-progressive challenge” and “genuine active involvement” are more effective motivational resources than interestingness. In addition, interestingly, S-students appreciated the dance choreography representation more than H-students.

These results harmonize with the findings of game-based learning research, which also emphasizes the importance of the challenge factor. A recent study performed by Hamari et al. (2016) concludes that challenge could be an especially strong predictor of learning outcomes. Hung, Sun, and Yu (2015) investigated whether challenging games are more able than matching games to improve students’ motivation. They report that students involved in the challenging games

achieved better flow experience (Csikszentmihályi, 1990), learning performance, and satisfaction.

With respect to the motivational role of the “genuine active involvement”, we observed that the “doing phases” of the learning process, besides unmasking false sensations of understanding (especially in the case of H-students), revealed students’ misconceptions and helped them to really grasp the logic of strategy. As a consequence, students became more motivated to succeed.

Previous research in the topic of active learning also revealed that meaningful interactivity could significantly contribute to positive attitude, quality of learning, and motivation (Grigorovici, Nam, & Russill, 2003; Sundar, Kalyanaraman, & Brown, 2003; Thorson & Rodgers, 2006; Evans & Gibbons, 2007). For example, Lumpkin, Achen, and Dodd (2015) report that the participants of their investigation particularly appreciated their active involvement in learning activities, highlighting that engagement positively impacted their learning.

In addition, our conclusion with respect to the role of “doing phases” in unmasking false sensations of understanding is in line with the statement Nemirow (1995) made regarding the phenomenon of understanding: “To understand is to be able to implement or apply a rule.” According to Wittgenstein (2009), someone who exclaims that “Now I can go on!” but fails to do so certainly did not understand it. Brantingham (2011) argues that the feeling that accompanies understanding (a belief that we have got the idea) should not be equated with true understanding. If this feeling is not followed by explanation or action, then understanding has not occurred.

We share the conviction of Grover and Pea (2013) that just as basic literacy in mathematics and the sciences is considered essential for understanding how our world works, AT/CT is just as essential in understanding how the all-pervasive computing devices work. Accordingly, a unified “CT for all” approach, akin to initiatives like “Science for all” or “Arts for all”, which is carefully designed and properly calibrated and which bridges the diversity gaps in order to target all learning communities, is a major endeavour. The studies we have presented should encourage curriculum developers and instructional designers to analyse the possibility of designing and developing unified AT/CT promoter learning environments for all students.

11 MULTIDIMENSIONAL EXPANSION OF THE ALGORHYTHMICS ENVIRONMENT

From 2016, new members have joined the AlgoRhythmic research group, and the enlarged team decided to expand the project. Four new algorithmic dances were created (heap sort, linear and binary search, backtracking), and the animation module of the web application was supplemented with several new elements.

11.1 Expanding the AlgoRhythmic collection

After analysing the feedback we had received through the AlgoRhythmic YouTube channel, three dimensions were identified by which the expansion took place. These followed the main concepts of the original idea: promoting CT by technologically and artistically enhanced multisensory tools.

11.1.1 From 1D view to 2D view

The first four sorting algorithms included in the AlgoRhythmic collection have quadratic time complexity ($O(n^2)$): bubble sort, insertion sort, selection sort, and shell sort (for details, see *Chapter 7*). These sorting strategies could be illustrated by easily comprehensible linear visualizations (see *figures 7.1–4*). The change of the array containing the numbers is visible after each step, and the sorting process is performed in a linear manner.

The next two algorithms were quick sort and merge sort. Their implementations are somewhat harder to visualize, and it could assume multiple views. Even *I Programmer* experts claimed that the quick sort choreography cannot be done using folk dances, and even with modern dances it would be a challenge to solve the visualization (*I Programmer*, 2011). These algorithms have $O(n \log n)$ average case time complexity and are considered more optimal than the first four in case of sorting random data. They apply the so-called “divide and conquer” paradigm, which is commonly visualized by a tree structure. Despite of this, these sorting strategies are perfectly understandable from the dance choreographies “applied” on linearly visualized arrays (see *figures 7.5–6*). After watching the AlgoRhythmic quick sort choreography, the above referred *I Programmer* expert stated: “Yes I know I claimed that it would be impossible, or if possible the result would be a modern dance the like of which we have not seen, but.... they have done it. ...two hats are used to mark the progress of the scan.”

Being motivated by user requests, the next visualization we designed was the heap sort choreography. This algorithm has $O(n \log n)$ time complexity too. The data structure used is the heap, a nearly complete binary tree, where each internal node has a greater (max-heap) or smaller (min-heap) value than any of its children.

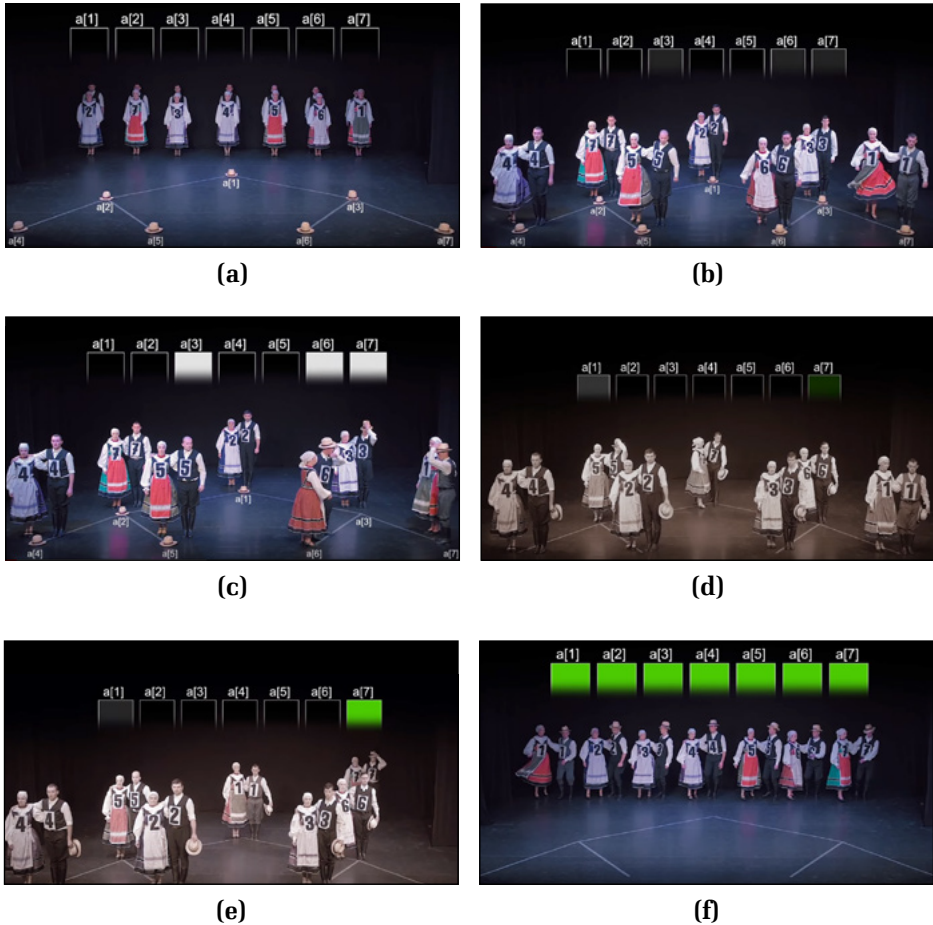


Figure 11.1. Key moments from the Heap sort AlgoRhythmics choreography: (a) the sequence to be sorted stored in a 1D array; (b) the sequence represented as a binary tree; (c) the development of the heap property begins (the dancers in the focus put on their hats, and their positions in the array are highlighted); (d) the heap has been constructed (all “parents” “are bigger” than their “children”); (e) the element that has reached its final position, returns to the array; (f) the heap-sorted sequence

The array to be sorted can easily be “transformed” into a binary tree using the formula: the first element of the array is the root, the child nodes of every internal node i are located (if they exist) in the array at the $[2i]$ and $[2i + 1]$ positions. When programming this method, the translation of the array positions into binary tree nodes is straightforward with the given formula.

We first tried to represent this algorithm linearly as well. Since we found the resulted choreography draft to be too complicated, we decided to explicitly visualize the tree structure and to illustrate the sorting strategy on this 2D scene (see *Figure 11.1*). At the beginning of the performance, the dance couples (representing the numbers) are arranged linearly as they are stored in the array. As a next step, the array opens into a binary tree representing the heap. The operation sequence of the algorithm is visualized on this 2D structure. As the numbers reach their final positions, the corresponding dance couples return to the array.

11.1.2 Including new dance styles

The first six choreographies are based on different folk dance styles, and the numbers are represented by individual dancers. In the case of heap sort, as mentioned above, we used a folk dance (“Mezőségi”) choreography, in which the numbers are represented by couples (see *Figure 11.1*).

Besides, based on the large number of foreign views (from other regions than Transylvania), we decided to internationalize the music and the choreography of the demonstrative dances. This way, a new style was introduced, where algorithm visualizations make use of the rhythm and style of the well-known southern Spanish flamenco (see *figures 11.2–3*). Flamenco was declared by UNESCO one of the “masterpieces of the oral and intangible heritage of humanity” (UNESCO, 2010). This art form fits perfectly into the cultural-artistic framework of our project since, according to the criteria for inscription into UNESCO, “it is strongly rooted in its community, strengthening its cultural identity and continuing to be passed down from one generation to the next”. It can also be considered that there was a dual change as flamenco is not exactly a folk dance, and it is from a different region.

To increase the diversity of the AlgoRhythmic collection, the last element we added illustrates the algorithm by a professional ballet choreography (see *Figure 11.4*).

11.1.3 Moving on to a new algorithm family

As sorting algorithms were covered by the first seven choreographies, we decided to move on to another very commonly used algorithm family, the searching algorithms. These algorithms can be classified based on their mechanism of searching: linear, binary, and backtracking search.

Linear search algorithms check, successively, every element of a list for the one that equals the value we are looking for (target key). Binary searches repeatedly target the middle element of an ordered list reducing the search space by half after each step. We have chosen to illustrate the linear and binary searching strategies by flamenco dance choreographies (see *figures 11.2–3*). The list to be searched could be represented by a flamenco dancer sequence (women) and the target key by an extra dancer (man; the protagonist of the performance). Since each dancer wears the corresponding value on his/her back, these are not visible in the case of list elements but are visible on the back of the target key.

In the case of linear searching, the result of the comparison operation between the target key and the current element of the searched list is yes (equal) or no (not equal). These yes/no comparisons could be illustrated by pieces of choreographies where the two dancers are dancing identically/differently (see *Figure 11.2*). In the case of binary searching, comparison operations result in one of the following three possibilities: less/greater/equal. To illustrate these variants, the corresponding dance couple could follow the same piece of choreography but in a slower/faster/in-synchrony rhythm (see *Figure 11.3*).

Backtracking can be seen as an advanced searching strategy. It is a programming strategy for finding all (or some) solutions to a given computational problem that incrementally builds solutions and immediately abandons all those partial solutions that evidently cannot be completed to a valid final solution. The classic backtracking example is the so-called “eight queens puzzle”, which asks for all valid arrangements of eight chess queens on a chessboard (no queen attacks any other). We chose to illustrate the recursive version of the four-queen variant of this classic backtracking algorithm by classic ballet choreography (see *Figure 11.4*). The following pieces of choreographies are attached to each ballerina:

- The queen comes to life on cell 0 of her row (new recursive call).
- The queen dies on cell 5 of her row (the current recursive call ends).
- The queen goes into “hibernation mode” (the current call is suspended; a new one begins at the next row).
- The queen comes back from the “hibernation mode” (the suspended call from the previous row continues).
- The current queen moves to the next cell of her row.
- The current queen successively considers those queens that hibernate on previous rows.
 - These ones temporally wake up for a “pair of mutually attacking”/“pair of mutually non-attacking” dance.

Additionally, four-queen “victory dances” illustrate the two valid solutions.

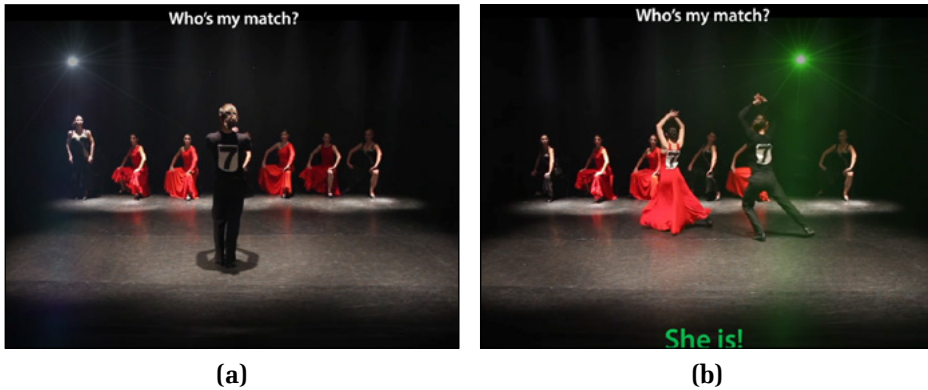


Figure 11.2. Key momentums from the Linear search AlgoRhythmic choreography: (a) the sequence $(x[0..6])$ to be searched;
 (b) his match is girl $x[5] = 7$



Figure 11.3. Key momentums from the Binary search AlgoRhythmic choreography: (a) the sequence to be searched; (b) the middle element is inspected; (c) subsequence $x[0]...[3]$ is eliminated from the searching space; (d) his match is girl $x[4] = 7$

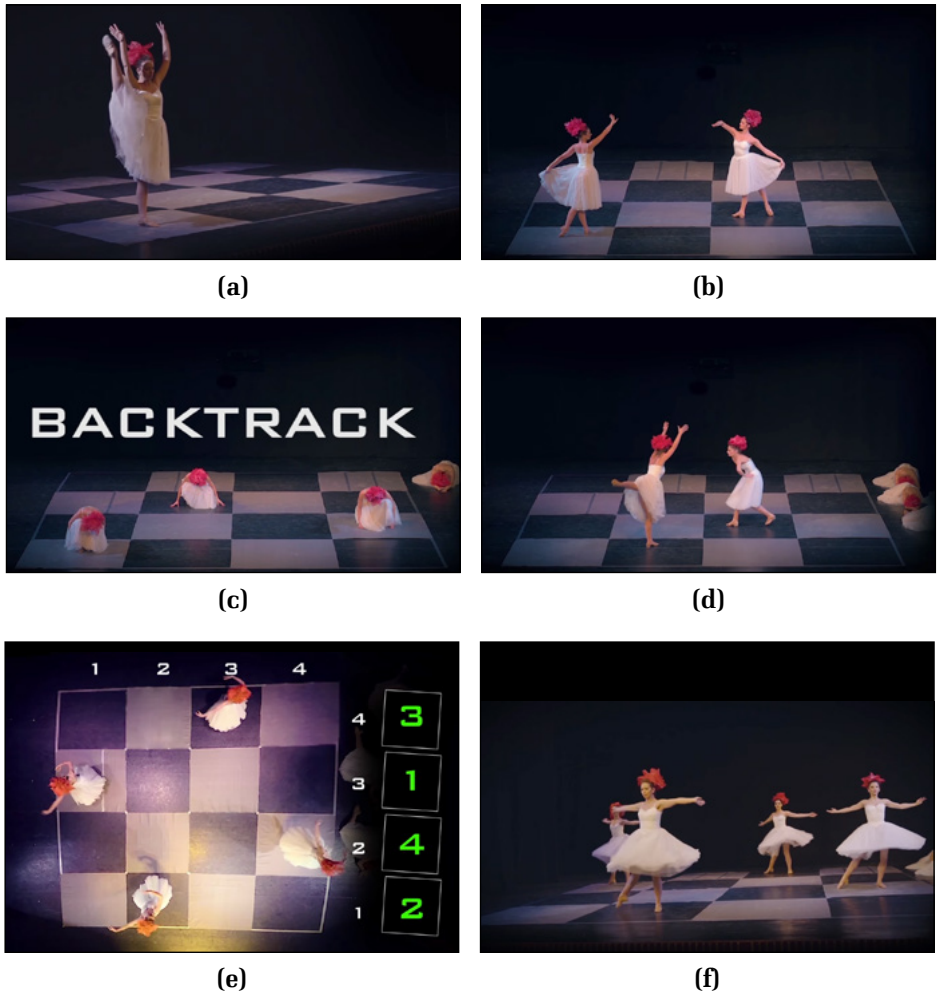


Figure 11.4. Key momentums from the Four queens AlgoRhythmic choreography: (a) introductory dance; (b) non-attacking scene; (c) queen 3 is going to come back from the “hibernation mode” (queen 4 is dead on cell [4][5]); (d) attacking scene; (e) first solution; (f) four-queen “victory dance”

11.1.4 Feedback

In Subchapter 7.1.1, we presented early (2011–13) evidences of the impact of the initial dance choreography collection (first six videos). In the following, we list some recent YouTuber comments (from the 2014–2020 period) for all ten videos. The first comment after we had uploaded our new videos in 2018 was: “Wow 6 years later this channel revived from their ashes!”

Most of the users who commented on the videos were CS students. They generally appreciated that the visualizations helped them understand and memorize the algorithms: “This helped me finally understand quick sort”; “Explained, without explanation, a lot of questions I had”; “I can no longer forget quick sort”; “I always go back to these videos when I need a refresh of sorting algos, they are so much easier to understand and remember than textbooks”. Some of the comments are about the students thanking us for helping them through the exam: “Thank you amigos, you saved my day, I passed my exam”; “Estos videos son geniales, me salvaron en un examen de programación”. Many users expressed regret that they did not learn the algorithms at school/university with these visualizations: “This is so much better than hours of lecture in my university”; “I wish my teacher explained sorting like this instead of being so boring”; “This is much clearer than any material I have ever encountered”; “I’ve never enjoyed studying for finals more. I wish my teacher had used this in class”; “If my teacher had shown these videos, the class would have been clearer”; etc.

On the other hand, there were non-CS majors too who expressed their appreciation: “Not a computer science major, don’t even know a thing about computers, but I watch a lot of those mesmerizing sorting videos. This is by far the best suggested video I’ve ever received”. One student wrote that even his parents understood the visualizations: “First time I found any sort of explanation for a sorting program that was simple enough for my egghead parents to understand”. Others showed their children the videos: “Cool way to show kids”.

We were particularly happy whenever we received comments from CS teachers: “I show every semester to my Data Structures & Algorithms students. It makes it SO much easier to talk about how it works”; “What a great implementation of an algorithm! I don’t even have to say anything to my students and they understand the algorithm”; “This is a fun way to watch it. I programmed ‘Sorting Out Sorting’ (30 minute movie - you can find it on YouTube) in 1979-81 (early days of computer graphics) and only really understood Quicksort after I saw what I’d programmed in graphic format (looks very similar to this – no Hungarian dancers however)”.

The majority of the comments are about the general impression of the users (“This Is One of the best things ever created by humans”), but there are a few that appreciate the solution we have chosen to visualize the algorithm. For example: “(Heap-sort) So this can be elaborated as Dance battle to get the top position in a tree only to give it to the lowest candidate in a hierarchic and settle out side

of the tree watching others to dance fight for the top position. Best explanation I could imagine”; “I like how the heaps are represented as a family”; “The most profound visual explanation of linear sorting that I have ever seen! The creativity i i i”; “(Quick sort) I can feel their bliss when each got both of the hats, found his place and can rest peacefully. (15 years of CS here)”.

Comments that emphasize the uniqueness and innovativeness of the visualizations are also common. For example: “I think I just found a new continent”; “Knuth needs to update Volume 3 with this”; “Art + Music + Logic + Sorting = This Awesome Video. Thank you very much”; “They really put the ‘rhythm’ in ‘algorithm’”; “They are Gems. Never imagined someone could come up with such creativity”; etc.

Some users draw attention to Hungarians in the comments: “That is why John von Neumann was born in Hungary”; “This is why Hungary has the highest number of no[b]el prize winners per capita”; “Brings a whole new meaning to “Hungarian notation””; “I believe that this should now be named as the ‘Hungarian Sort’”; “I am a software developer listening Hungarian Rhapsody No.2 and YouTube recommends me this. LOL!”; “Thank you to the YouTube algorithm for deciding to show me this video! I studied mathematics for a month in Budapest in 1998. This is beyond words so fantastic”; “This and kürtös kalács are the only reasons why God invented Hungarians”.

Evidently, not all comments are positive. Most of these blame the videos for being too long. For example: “(Quick sort) Wow, even slower than bubble sort, Hungarian folk dancers just aren’t a good architecture for running sorting algorithms”; “No multi-threading? Come on, optimize your dancers already!”; etc.

The selection below is intended to help readers perceive more directly the taste and the aroma of appreciating comments:

- “Most beautiful thing I have ever seen in programming teachings”; “Terrific Love the Overall Concept”; “This is so satisfyingly systematic”; “Good idea for effective learning”; “Well demonstrated! Unconventionally, but well”; “Creativity level: infinite”; “Excellent! Great job, very good idea for illustrating an abstract thing”; “Now that’s what I call interpretive dance!”; “This is the way we should learn about serious things in Computer Science. Awesome videos”; “So didactic”
- “30 years working in IT, this is clearly the highlight; This is no joke the best learning tool for these algorithms I’ve ever seen”; “Was 5 minutes ago watching sorting algorithm videos wondering what was happening? I now understand”; “Forget about theoretical explanations... this is the best way to understand sorting algorithms”; “Ok, thank you. I had a hard time trying to understand shell sort, here I understood literally after 30 seconds”; “Never thought that this algorithm is understandable”; “Amazing! Makes it so easy to understand the whole process and, of course, is a funny way to get into another countries’ folk dance. Congrats for the awesome work!”

- “I can’t express how much I adore you! I’m studying computing 4 years already, and every time I have to write some sorting algorithm, I firstly check out your dance!”; “I would always come back here when I forget about sorting haha”
- “Literally, the best thing ever. Thanks so much Romania. You’ve done the world a favour”; “Of all the videos I’ve seen about quicksort, I actually finally learned it from this. Thanks Hungarian dancers!”; “I don’t know who came up with this idea but it’s so much easier to understand sorting algorithms thanks to these Hungarian folk dances”
- “This is art”; “This is sooo great - Art meets Science”; “Beautiful piece of art to explain Computer Science, this is so EPIC!”; “Awesome, helpful and creative. Way more fun to review sorting algorithms with folk dancing”
- “Ideally, that’s how teaching should be in general. Entertaining and informative. But few teachers have enough passion, time and are paid well enough to do so...”
- “Very nice performance!! This area where art merges with learning, is amazing!!! It’s just that it needs some good marketing”
- “This Video made me understand my informatics class, but now I want to be a dancer”
- “Whaw, that was pretty amazing. This changed my life forever. I am not the same person anymore, just thank you”
- “Best thing ever... top candidate”; “Super! You are the best professors”

11.2 From dance to code

For many years, dance choreographies have been the centre of the AlgoRhythmic universe. As a next step, we decided to redesign the animation module of the AlgoRhythmic web application (see *Figure 11.5*). As a result, a completely new interactive learning environment was created in the form of an intuitive software which guides students in the development of algorithmic thinking through multifarious levels of interactivity (Nagy, Osztián, Cosma, Kátai, & Osztián, 2019 – Looking for the Optimal Interactivity Level in the AlgoRhythmic Learning Environment. In: *EdMedia+ Innovate Learning* (pp. 106–114). Association for the Advancement of Computing in Education (AACE)). This platform can be seen as a “dance floor” where the “choreographer” can predefine courses and can specify the level of user interaction. As the “dancers” pick up the “rhythm of the algorithm”, the possibility of controlling it will be also given to them. The code of the algorithm will also appear on the “scene”, being built by the user, and then “become alive” by being executed together with the animation. It is like leading students from the dance choreography to the code. For this purpose, they are guided through some steps which are the main elements of the renewed learning environment.

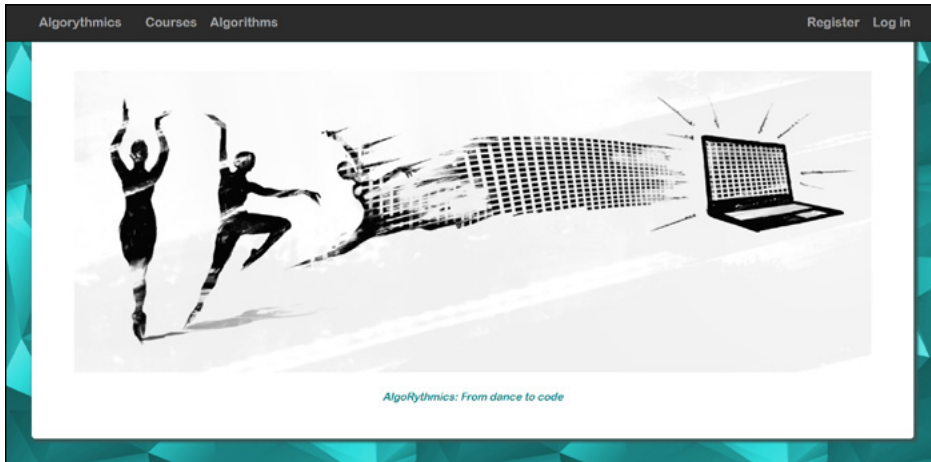


Figure 11.5. *The redesigned AlgoRythmics web application*

11.2.1 Learning steps

Video: The algorithms are represented visually by dance choreographies. These videos can be automatic or interactive. In the first case, the user is able to visualize the whole algorithm without any interruption, while in the second case the video is stopped, waiting for the user to answer a question.

Animation: A step from the dance towards the code is the animation, which is a more abstract level of the algorithm (see *Figure 11.6*). The animation can be also automatic or interactive. The interactive animation is paused in some key moments – predefined by the administrator/teacher –, and it continues to play if the user provides the operation that should happen next in the flow. These operations are, for example: selection (when the algorithm iterates and further elements will be analysed), compare (when two elements must be compared), and swap (in case two elements have to be swapped). Getting familiar with these functionalities, the “in control” step will be easy to understand.

In Control: Taking the user experience to a new level, we give control to the user. Using previously gained knowledge of a selected algorithm, the user has the possibility to manually conduct the entire algorithm, using the previously mentioned operations (for example): select, compare, or swap. Like its predecessor, this module also operates in both white- and black-box modes.

Create Code: This is the learning step when the user is obliged to use their listening for identifying the loop structure of the selected algorithm (single loop, consecutive loops, or imbricated loops) based on sound effects. Furthermore, this is the phase in the learning process when the user finally arrives at the point where the code must be written (see *Figure 11.7*).

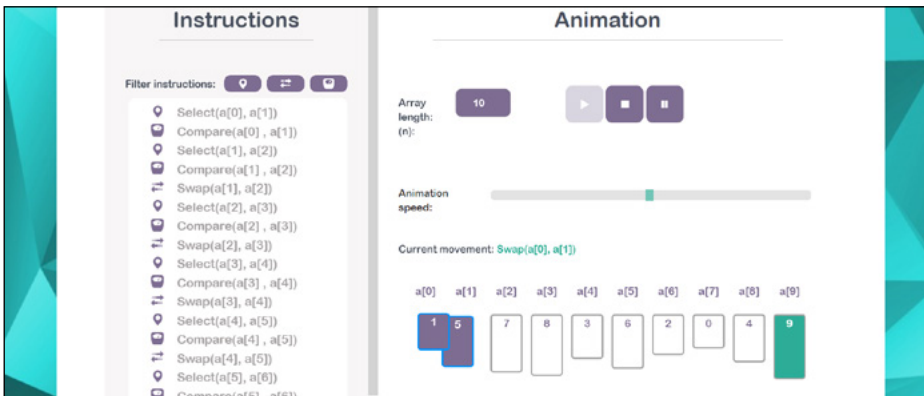


Figure 11.6. Animation module

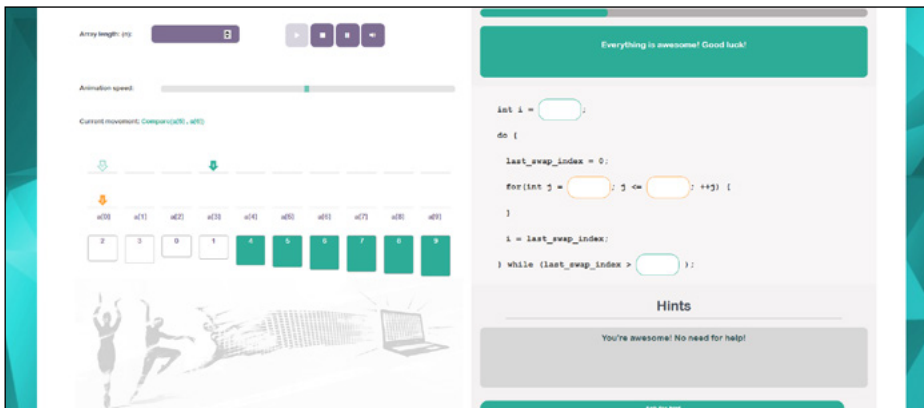


Figure 11.7. Create Code module

Code Comes Alive: Summarizing all the previous elements, in this phase, the user can pay attention to the result of their work: parallel with the step-by-step execution of the code (the relevant code lines are highlighted), a synchronized animation is played.

The learning steps described above have the purpose of helping in the acquisition of algorithmic knowledge. For verifying the efficiency of this process, we introduced the notion of the tests in the renewed tool. These can be defined very dynamically and will be evaluated automatically after submitting.

11.2.2 Courses

The role of the courses is to help users understand algorithms. Different combinations of learning steps, in many variations, create different courses.

The learning steps that a course is built with can be differentiated by their input, display, and playback type. These can be defined dynamically by the administrators or teachers, with the possibility of managing the level of user interaction. During these courses, users encounter difficulties, which could result in making mistakes or requesting help. The software registers all these events.

11.2.3 Levels of interactivity

The renewed environment operates on three levels of interactivity.

- 0 interactivity (no interactivity): the user is an independent observer; s/he can attend to the animation and the video without any interruption;
- $\frac{1}{2}$ interactivity (half interactivity): the user is partially involved; at some specific key moments, s/he needs to answer some questions or continue the animation flow;
- 1 interactivity (full interactivity): the user needs to control the whole animation; s/he is the conductor of the algorithm, the embodiment of operations.

In the next chapter, we will focus explicitly on this new feature of the AlgoRythmics environment.

11.3 Promoting computational thinking in the extended AlgoRythmics environment

In their endeavour to define CT, the International Society for Technology in Education (ISTE, 2020) and the American Computer Science Teachers Association (CSTA, 2020) identified nine related concepts suitable to be promoted within the framework of K–12 education: data collection, data analysis, data representation, problem decomposition, abstraction, algorithms, automation, parallelization, and simulation. ISTE also adds an operational definition for CT as a problem-solving process with characteristics such as (ISTE, 2020):

- formulating problems so their solutions can be represented as computational step sequences (skill 1);
- logically organizing and analysing data (skill 2);
- representing data through abstractions (modelling, simulations) (skill 3);
- automating solutions through algorithmic thinking (representing them as algorithms) (skill 4);
- identifying, analysing, and implementing possible solutions with the goal of introducing concepts such as algorithm time and space complexity (skill 5);

- generalizing and transferring the studied problem-solving process to a wide variety of problems (skill 6).

The authors of a study (Mannila et al., 2014) stress that practising and developing CT-related skills is not limited to CS/STEM subjects. On the other hand, they argue that programming can be seen as a tool for developing all CT aspects if we clearly distinguish between the concepts of programming and coding. They suggest that coding should be seen as the last phase of the multifaceted programming process when the solution that has been achieved through such previous phases as analysis, decomposition, and design is implemented. Accordingly, while AlgoRythmics environment has the potential to promote CT for all students, it generates a learning experience that could be a prelude to the coding phase for those who are interested in it.

11.3.1 Shifting to blended learning

The studies we presented in the previous chapters confirmed the potential of the AlgoRythmics environment to support users in assimilating the strategy of the analysed algorithms (*skill 4*). Our feedback from middle/high-school and university-level teachers also confirmed that they encourage self-paced learning mode mostly with the goal of discovering the strategy of the algorithm (YouTube comments support this aspect). This means – as we are going to show it in the following – that the potential of the learning environment to promote CT is only partially exploited. According to its ISTE/CSTA definition presented above, promoting CT implies more than understanding and performing algorithms, and we were taking this fact into account during the designing process. To improve these shortcomings, we suggest a shift to the principles of blended learning.

Blended learning can be defined as an optimal mixture or the effective integration of various learning techniques, technologies, and delivery modalities to maximize the learning outcome (Singh & Reed, 2001; Valiathan, 2002; Finn & Buccheri, 2004). In line with this approach, we suggest in-person classroom activities facilitated and complemented by a teacher: while the students are studying the algorithms in the above described e-learning environment, the teacher guides their learning experiment to strengthen it (using techniques such as effective questioning). For example, the teacher can draw students' attention to the way the principle of sequenced multiple representation was implemented: the number sequence is personified by a dancer sequence, stored in white and black arrays, and represented as colour bars (*skill 3*). Key attributes of multiple representations (complementarity and redundancy) should also be identified (Meij & Jong, 2006). In the following, we suggest other ways how teachers can provide CT support for students (during their journey with the AlgoRythmics learning tool).

Using questions is an effective way to stimulate the recall of prior knowledge, promote comprehension, and build critical thinking skills. Well-worded questions can stimulate students to think about a topic in a new way. Effective teachers are able to formulate questions to fit the cognitive level of the students (Hattie, 2012). Other characteristics of the effective use of questions are: careful phrasing and word clarity; creating psychologically safe environment; appropriate sequencing and balance; properly calibrated wait time (Tofade, Elsner, & Haines, 2013).

11.3.2 Exploring searching strategies from an algorithm complexity perspective

After students have successfully orchestrated the studied searching algorithms, they are prepared (the required psychologically safe environment has been created) for analysing them from the perspective of complexity (*skill 5*). It is plausible, for example, to assume that they will be able to answer the below questions regarding best and worst cases with respect to searching algorithms. Additionally, as we mentioned above, students can also ask the software tool to generate best/worst case inputs for the searching/sorting algorithms.

We suggest the following question sequence:

- How many comparison operations does the linear search algorithm imply for the sample presented in *Figure 11.2*?
- How many comparison operations does the binary search algorithm imply for the sample presented in *Figure 11.3*?
- What is the “best case” with respect to the linear search algorithm? (The value we are looking for is located in the front of the list; only one comparison is needed to be found.)
- What is the “worst case” with respect to the linear search algorithm? (The value we are looking for is missing from the list.)
- What is the “best case” with respect to the binary search algorithm? (The value we are looking for is the middle element of the list; only one comparison is needed to be found.)
- What is the “worst case” with respect to the binary search algorithm? (The value we are looking for is missing from the list.)

If students possess the necessary mathematical knowledge, the teacher should continue with the following questions (the graphical representation of the corresponding functions could also be illustrative):

- How many comparison operations does the linear search algorithm imply (for a list with n elements) in the “best case”?
- How many comparison operations does the linear search algorithm imply (for a list with n elements) in the “worst case”?

- The worst case time complexity of the linear search algorithm is $O(n)$. Why? (We do not encourage detailed discussion regarding big O notation.)
- How many comparison operations does the binary search algorithm imply (for a list with n elements) in the “best case”?
- How many comparison operations does the binary search algorithm imply (for a list with n elements) in the “worst case”?
- The worst case time complexity of the binary search algorithm is $O(\log n)$. Why?

[favourable moment for introducing the concept of binary tree]

Another variant for students with less mathematical knowledge:

- Given the numbers from 1 to 1,000, what is the minimum/maximum number of guesses needed to find a specific number if you are given the hint “guessed”/“missed” (linear search) or “guessed”/“higher”/“lower” (binary search) for each guess you make?

The teacher might also guide students in discovering the reasons behind the superiority of binary search strategy against the linear one. A possible approach (that also promotes *skill 2*) is: the length of a list with n elements versus the height of a binary tree with n nodes. The following questions allow the teacher to introduce the concept of search space, too. In the essay entitled *Thinking about computational thinking*, the authors suggest methods for familiarizing grade 3 students with this concept (Lu & Fletcher, 2009).

- How many elements of the search space are eliminated by each try (comparison operation) in the case of linear/binary search algorithm?

(Another perspective: with each try, the current problem is reduced to searching in one shorter vs. half shorter list.)

Since in the worst case binary search scenario the algorithm continues with the larger part of the divided sequence, the teacher can help students realize that starting with the middle element is an optimal choice (the larger part of the search space is minimal).

After students have studied sorting algorithms too, it can be analysed when the “sorting + binary search” strategy is preferable instead of linear search.

According to our experience, the above-presented approach has the potential to support students in realizing the relative value of the studied algorithms.

11.3.3 Exploring sorting strategies from an algorithm complexity perspective

We suggest the following question sequence to be analysed:

- How many comparison and swapping operations does the selection, bubble, insertion, and merge sort algorithm imply for the samples presented in the videos from figures 7.1, 7.2, 7.3, and 7.5?
- What is the “best case”/“worst case” with respect to a sorting algorithm?

(The input sequence is already sorted in increasing/decreasing order; it is plausible to assume that students will be able to comprehend this intuitive fact.)

If students possess the necessary mathematical knowledge, the following questions can be discussed for arbitrary length sequences (n), otherwise for a specific length (for example: 10, the length of the dancer sequence). Complexity analysis can be supported by introducing the concept of problem decomposition (another key ability regarding CT; see the ISTE/CSTA definition of CT): sorting by successive selections/bubbling/insertions.

- How many comparison and swapping operations does the selection, bubble, and insertion sort algorithm imply (for a list with n elements) in the “best case”?
- How many comparison and swapping operations does the selection, bubble, and insertion sort algorithm imply (for a list with n elements) in the “worst case”?
- The best case time complexity of the selection sort algorithm is $O(n^2)$. Why? (It is probably enough to transmit the idea that the number of the needed basic operations (only comparisons) is $(n(n-1))/2$, which is a second-degree function.)
- The worst case time complexity of the selection sort algorithm is $O(n^2)$. Why?

(The number of the needed basic operations (each comparison is followed by swapping) is $(n(n-1))$, which is also a second-degree function.)

- The best case time complexity of the bubble and insertion sort algorithm is $O(n)$. Why?
- The worst case time complexity of the bubble and insertion sort algorithm is $O(n^2)$. Why?

The teacher should guide students in realizing why some “best case” complexity values differ and why the “worst case” ones do not.

Before analysing the time complexity of merge sort algorithm, the concept of divide and conquer should be introduced. This approach will guide students again to a binary tree. To simplify the calculus, we suggest that $n = 2^k$.

- How many comparison operations does the merge sort algorithm imply (for a list with n elements) in the “best case”?
- How many comparison operations does the merge sort algorithm imply (for a list with n elements) in the “worst case”?

Students only need to understand that at each level of the $\log(n)$ height binary tree the merging processes assume $O(n)$ basic operations.

After these introductory analyses, even the complexity analysis of shell, quick, and heap sort algorithms can be addressed in the same manner.

Besides this complexity analysis, the parallel simulation of the algorithms (as they are working side by side on different colour scale bars; see *Figure 7.7.f*) also supports students in realizing the relative value of them (*skill 5*).

The merge sort (and quick sort) dance video offers an excellent opportunity for introducing the concept of parallelization too, which is also included by ISTE/CSTA (2020) among the CT-related key abilities. “Why boys stand idle while girls...?” is an obvious question also suggesting the need for parallelization.

11.3.4 Exploring backtracking strategies from an algorithm complexity perspective

Time complexity aspects of n-queens backtracking algorithms can be suggested by discussing the following questions (see *Figures 11.4*):

- How many possibilities are to arrange 4 pawns on a 4×4 chessboard if the only restriction is: no pawns in the same row? How many pawn ballerinas are needed for a four-pawn backtracking choreography?
- How many possibilities are to arrange 4 rooks on a 4×4 chessboard in such a way that no rook is positioned to attack any other rook (no rook pairs in the same row or column)? How many rook ballerinas are needed for a four-rook backtracking choreography?
- How many possibilities are to arrange 4 queens on a 4×4 chessboard in such a way that no queen is positioned to attack any other queen (no queen pairs in the same row, column, or diagonal)? How many queen ballerinas are needed for a four-queen backtracking choreography?

If students master the related mathematical apparatus (exponential function, permutations), the teacher might continue as follows:

- How many possibilities are to arrange n pawns on an $n \times n$ chessboard if the only restriction is: no pawns in the same row?
- How many possibilities are to arrange n rooks on an $n \times n$ chessboard in such a way that no rook is positioned to attack any other rook?
- Although implementing the n-pawns backtracking algorithm is easier than implementing the n-queens one, why is “the valid n-queens arrangements are such valid n-pawns arrangements where no pawns are in the same column or diagonal” approach inefficient?
- Usually, backtracking algorithms have exponential time complexity. Why?

11.3.5 Basic characteristics of algorithms: Generality

According to the ISTE definition, CT implies the skill of generalizing and transferring the studied problem-solving process to a wide variety of problems (*skill 6*). In line with this, the teacher should help students notice that the studied algorithms/searching strategies are applicable to any sequence (or any sorted sequence in the case of binary search).

- The protagonist of the performance is searching for a dancer sequence wearing the numbers on their back.

- The learning environment makes possible the algorithm orchestration processes (for both searching and sorting algorithms) on random, invisible sequences.

11.3.6 Computer algorithm “optimization”

Promoting CT assumes sustaining students in distinguishing between tasks formulated to be performed by human beings versus tasks formulated to be processed by computers (*skill 1*). This is relating to the concept of optimizing computer algorithms versus human-processed algorithms. In Kátai (2014a), an example is presented on how teachers can make use of the bubble-sorting dance choreography in this sense.

Table 11.1 shows the sequence to be sorted after each pass (the elements are indexed from 0 to 9). After each pass, dancers who had reached their final position (at the end of the sorted list) turned back (bolded elements). Currently, bolded elements are underlined, too. In the first pass (0..9), the last swapping operation was swap (a[7],a[8]) (grey cells) and, consequently (according to the optimized version of the algorithm), both dancers, 8 and 9, turned back. After the second pass (0..7), by chance, also two elements (6 and 7) were on their final places, but only one of them (number 7) in a proven way (the last swapping was: swap (a[6],a[7])). Consequently, only dancer 7 turned back.

A typical question posted by many YouTubers was:

User1: Why did 6 vs. 7 not both turn back but 8 vs. 9 and 5 vs. 6 did?

One of the users answered this question in the following way:

User2: There is a memory of where the last exchange took place, so all values greater than the memory must be in the correct order, or there would have been a change. For 6 and 7, there is no proof that they were already in order because the change took place at the last comparison, so the memory was only of the value in the highest index.

Since user 1 (as a human being) saw that after the second pass numbers 6 and 7 reached their final places, this tended to shorten the next pass by two elements. On the other hand, user 2 comprehended that a bubble-sort-algorithm-guided blind computer “cannot realize” (after the second pass) that cell a[6] stores the right number.

Questions we suggest to be discussed with regard to the Bubble sort AlgoRhythmic video (see also *Figure 7.1*):

- How many elements reach certainly their final positions (in the ordered list) after each traverse the bubble sort algorithm performs?
- Why did two dancers turn back after the first pass?
- Why did only one dancer turn back after the second pass?

Table 11.1. *The state of array $a[0]...[9]$ after each pass of bubble sort algorithm*

	0	1	2	3	4	5	6	7	8	9
Initial sequence	3	0	1	8	7	2	5	4	6	9
After first pass	0	1	3	7	2	5	4	6	8	9
After second pass	0	1	3	2	5	4	6	7	8	9
After third pass	0	1	2	3	4	5	6	7	8	9
After fourth pass	0	1	2	3	4	5	6	7	8	9

12 ALGORITHM VISUALIZATION ENVIRONMENTS: CAN AN OPTIMAL INTERACTIVITY LEVEL BE ESTABLISHED? (STUDY 8)

One of our central goals in redesigning the AlgoRythmics environment was to increase user engagement in the algorithm visualization (AV) process. Consequently, one of the most important features of the renewed web application is its interactive character.

Relevant research in the field of AV reports mixed results regarding their educational value. One of the most important studies on AV effectiveness (Hundhausen et al., 2002) concludes that the way students use visualizations is more important than the visualizations themselves. According to Shaffer et al. (2010), an important conclusion from the literature is that to make AVs pedagogically useful – they must support student interaction and active learning.

As we described in the previous chapter, the expanded AlgoRythmics environment incorporates dance choreography illustrations and interactive abstract animations for ten computer algorithms. The animations include the so-called interactive prediction feature on three levels: “no interactivity” (passive viewing), “half interactivity” (the animation stops at predefined key moments and users have to indicate the next movement/operation) and “full interactivity” (users have to orchestrate the algorithm by interactively predicting the entire operation sequence). The study we are going to present in this chapter (Osztíán, Kátai, & Osztíán, 2020) focuses on the influence that varying degrees of interactivity have upon students’ learning. We analysed this issue with respect to shell sort algorithm. The experiment was conducted with first-year undergraduate students with different levels of prior knowledge in programming.

12.1 Different levels of engagement with algorithm visualizations

Interactive visualization has been employed in CS education since the 1980s. In 2002, Naps et al. (2002) reported a strong perception among educators that visualization can help (almost all respondents for their survey agreed with the statement “Using visualizations can help learners learn computing concepts”). Interestingly and contradictorily, in the same year, a meta-analysis (Hundhausen

et al., 2002) reported mixed results regarding the pedagogical benefits of visualization technology. However, Naps and his co-authors reanalysed the twenty-one experiments included in the meta-analysis performed by Hundhausen et al. (2002) and noticed that ten out of twelve (83%) experiments using learner engagement yielded a significant result. On the other hand, in the case of those nine experiments that manipulated representations, only three (33%) produced a significant result. Based on this observation, Naps et al. (2002) state that these results suggest that what learners do, and not what they see, may have the greatest impact on learning. In other words, they conclude that AV technology is of little educational value unless it engages learners in an active learning activity.

In addition, Naps et al. (2002) propose an engagement taxonomy including six different forms of learner engagement with AV technology: 1) no viewing, 2) viewing (student passively views an AV), 3) responding (student responds to questions about the content while viewing an AV), 4) changing (student changes the AV by, for example, providing input data to the algorithm), 5) constructing (student constructs an AV), and 6) presenting (student presents an AV to others) (Dani, 2016).

In the study performed by Grissom, McNally, and Naps (2003), the authors compare the performance of three treatment groups having different levels of engagement with visualizations: no viewing (not seeing any visualization), viewing (simply viewing visualizations for a short period in the classroom), and responding (interacting directly with the visualizations for an extended period outside the classroom). These authors also conclude that learning increases as the level of student engagement increases.

In the present study, we have included three levels of engagement: viewing, responding, and constructing. To implement the responding level, we applied the interactive prediction method. Regarding the constructing level of engagement, Karavirta and Shaffer (2015) note that a variation on this approach gives a data structure and an algorithm and expects the student to simulate the algorithm. In other words, students are invited to imitate the steps of an algorithm by manipulating the interface to control the progress of the AV. In the following, according to the “AlgoRythmics terminology”, we will identify these three conditions as no interactivity, half interactivity, and full interactivity.

Studying learning with dynamic visualizations that incorporate different interactivity levels is a multifaceted research area including intertwined factors. For example, two important principles that relate to the interaction design of dynamic representations are learner control segmenting and learner control pacing (Plass, Homer, & Hayward, 2009). It can be observed that half interactivity implicitly generates segmentation, and full interactivity implicitly results in learner control of pacing. In addition, from a learning perspective, we can distinguish between functional interactivity and cognitive interactivity (Plass et al., 2009). While an investigation focuses on cognitive interactivity aspects,

the analysed condition might differ from a functional interactivity perspective. More generally, the presence of a combined effect of several interdisciplinary factors (psychological factors, animation design, learning environment, didactic implementation, etc.) could offer a plausible explanation as to why results in this field of research are contradictory (as mentioned above).

12.2 Different levels of engagement in the AlgoRhythmic environment

The five steps around which the AlgoRhythmic environment is built (video, animation, in control, code building, code comes alive) include all six levels of engagement. As for the animations, these work in three operating modes: viewing, responding, and constructing. In viewing mode (no interactivity), students are independent observers, and the following basic user options are available to them: play, pause, and stop buttons and an animation speed slider (see *Figure 12.1*). In responding mode (half interactivity), students are partially involved, which means that at predefined key moments the animation suddenly stops, and user interaction is needed. In constructing mode (full interactivity), learners need to orchestrate the whole animation process: they are the constructors of the algorithm, the embodiment of the operations.



Figure 12.1. *Viewing mode (no interactivity)*

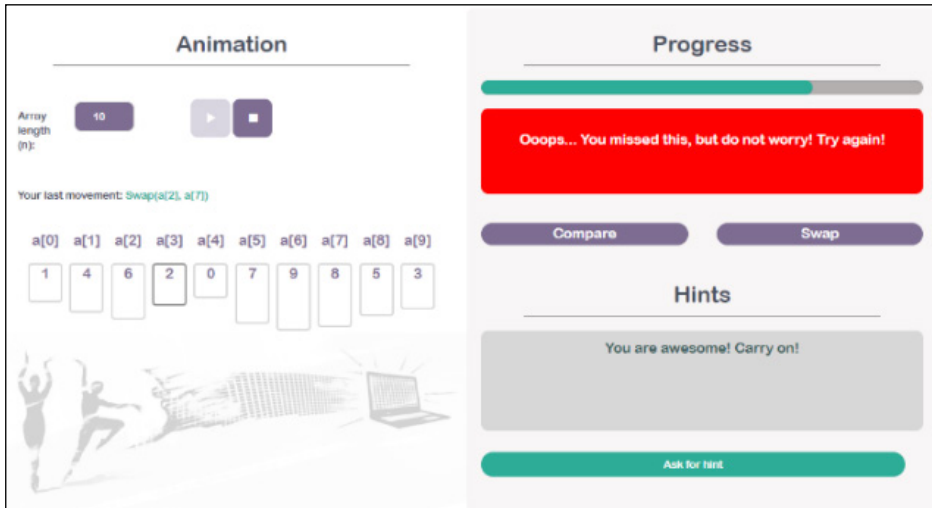


Figure 12.2. *Responding (half interactivity) and constructing (full interactivity) modes*

In order to choose the next correct movement, the following user options are available in both responding and constructing modes: select the pair of elements, compare and swap (see *Figure 12.2*). If users are unsure regarding the next operation, the environment provides the opportunity to ask for help (hint).

We assigned each interactivity level to a specific group: G1 (no interactivity), G2 (half interactivity), and G3 (full interactivity). All participants were able to complete each step once, and the input was a predefined sequence of numbers given by a teacher.

Based on the literature review, we hypothesized that learning would increase as the level of student engagement increases, and we addressed the following particular research questions:

1. RQ1: Does increased student engagement benefit the learning outcome (results in higher post-test scores)?
2. RQ2: How does students' prior programming knowledge influence comprehension?
3. RQ3: Is there a significant difference between female and male students' performance?
4. RQ4: Is there a correlation between the level of interactivity and the nature of the acquired knowledge?

12.3 The experiment

We designed a three-phase experiment (pre-test, learning phase, and post-test), and the investigation took place at Sapientia Hungarian University of Transylvania at the beginning of the 2019/2020 academic year.

One hundred and thirty-four subjects (14% females) participated in the study (first-year undergraduate students from five different educational programmes: Informatics, Computer Science, Automation, Mechatronics, and Mechanical Engineering). Participants were assigned to one of 3 categories based on their prior programming experience: No prior knowledge (NP: 0 years of high school programming experience), Basic prior knowledge (BP: 1, 2, or 3 years of high school programming experience in natural science classes; one to two classes per week), and High prior knowledge (HP: 4 years of high school programming experience in informatics classes; five to seven classes per week). Students from each category were randomly assigned to the three groups: G1, G2, and G3.

During the learning phase and the post-test, 46 students participated in G1 group and 44 students participated in both G2 and G3 groups (the three eliminated participants belonged to groups G2 and G3). The mean proportion of females was 10.86%, 18.18%, and 13.63%, yielding a non-significant difference based on Chi-Square test ($p = 0.6 > 0.05$). The mean of prior programming knowledge was 2.17, 2.11, and 2.20 (years) for the specific groups (G1, G2, and G3). Testing this proportion, we came to the same result: no significant differences (Chi-Square: $p = 0.98 > 0.05$) (see *Table 12.1*).

Table 12.1. *Proportion of students by prior programming knowledge and gender*

Total: 134 students		G₁	G₂	G₃
Gender	Male	41	36	38
	Female	5	8	6
Prior programming knowledge	NP	14	13	19
	BP	15	11	18
	HP	13	13	18
Mean of prior programming knowledge		2.17	2.11	2.20

Source: Osztián, Kátai, & Osztián, 2020

The preliminary questionnaire, the pre-test, and the post-test were conducted in the Socrative online classroom application. The preliminary questionnaire consisted of 8 questions. In order to test participants' CT, the pre-test included 8 questions based on programming-free tasks selected from the website of the Bebras (2020) contest. The learning phase was built around the AlgoRythmics animation of the shell sort algorithm (on 10-length sequence) since only 7% of the participants had heard about this sorting strategy and none of them had studied it. The experiment included the animation in all three modes: viewing, responding, and constructing. The post-test questionnaire consisted of the following 12 questions (potential range: 0–12):

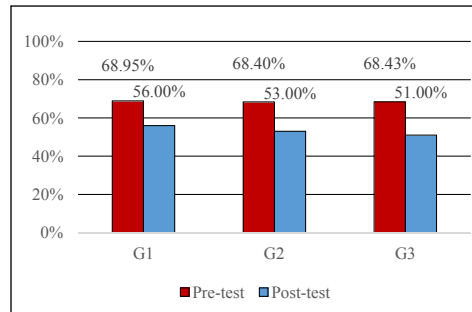
- Q1–6: Considering a “3-1 Shell sort strategy” (for gap values 3 and 1) on the sequence $x[0] \dots [6] = \{1, 19, 7, 8, 12, 11, 9\}$, which are the first three steps (indicate the operation and the pair of elements to be swapped or compared)? FIRST/SECOND/THIRD ($x[?]$, $x[?]$)
- Q7–8: Consider the “3-1 Shell sort strategy” for the sequence $x[0] \dots [6] = \{1, 19, 7, 8, 12, 11, 9\}$. After the compare ($x[3]$, $x[6]$) / compare($x[5]$, $x[6]$) operation, which is the next pair of elements to be compared?
- Q9–12 (generalization): How many compare/swap operations are performed by a “3-1 Shell sort” algorithm for a 7-length increasing/decreasing ordered sequence?

The experiment lasted a total of 2 hours. The pre-test was held for all three groups together in an amphitheatre of the university. As a preparation for the learning phase, we presented the AlgoRythmics online learning tool because many of the students were encountering this online learning environment for the first time (15 minutes). After this, each group (G1, G2, and G3) was assigned to a computer laboratory, where the learning phase began. During this, students were required to register for the online learning environment, after which they started to complete the designated course. For each group, the learning phase included two viewings of the animation. As a first step, all groups watched the animation in viewing mode (the core form of engagement) (Naps et al., 2002). During the second viewing, groups G1, G2, and G3 watched the animation with no, half, and full interactivity respectively. At the end of the experiment, participants from all groups were invited to answer the post-test questions.

12.4 Results and discussion

The pre-test performances (see *Figure 12.3*) were analysed with one-way analysis of variance (ANOVA). The independent variable was the instructional condition (no, half, full interactivity), and the dependent variable was the pre-test score (Levene's test showed that equal variances could be assumed: $p = 0.93 > 0.05$). As we expected, no significant differences were found ($F(2, 130) = 0.007$, $p = 0.99 > 0.05$).

As a next step, we analysed participants' post-test performance (see *Fig. 12.3*) with one-way analysis of covariance (ANCOVA). Again, the independent variable was the instructional condition, and the dependent variable was the post-test score (the assumption of homogeneity was met; $p = 0.27 > 0.05$). The pre-test performance was used as a covariate. Although a moderate decrease was noticed (G1: 56%, G2: 53%, G3: 51%), the differences were not significant ($F(2,130) = 0.846, p = 0.432 > 0.05$).



Source: Osztián, Kátai, & Osztián, 2020

Figure 12.3. *Pre-test and post-test results based on the level of interactivity*

This result does not support the assumption that learning will increase as the level of student engagement increases. In addition, it suggests that a universally optimal interactivity level cannot be established. Clearly, this conclusion is contrary to the majority of prior research in this field. For example, Shaffer et al. (2010) concluded, based on Hundhausen et al. (2002) and Naps et al. (2002), that a growing body of evidence indicates that the most important factor that contributes to the effectiveness of learning with AV appears to be engagement of the students' attention.

On the other hand, some previous results harmonize with our finding. For example, Jarc et al. (2000) also found no significant difference in educational outcomes for students who used the interactive version of their AV system (interactive prediction) compared to their peers who used the non-interactive version (passive viewing). Interestingly, Shaffer et al. (2010) use the expression of "engagement of the students' attention". It seems that watching an AV could be engaging even without any interactivity. In addition, Myller, Laakso, and Korhonen (2007) also report that they cannot confirm their hypothesis that ET level (engagement taxonomy: viewing and changing levels) would have a significant effect on the learning outcomes.

Similar results were found regarding the pacing functionalities of the animations. Prior research in this field also reported inconsistent findings. While

some studies showed a benefit of control (Mayer & Chandler, 2001; Schwan & Riempp, 2004; Boucheix & Guignard, 2005), others found no benefit (Adesope & Nesbit, 2012). One of the most important findings of the meta-analysis performed by Berney and Bétrancourt (2016) is that the positive effect of animation on static graphics was found only when learners did not control the pace of the display.

Our results suggest that all three levels of engagement we analysed may have both advantages and disadvantages. For example, a possible explanation of why interactivity may not always guide students to better learning outcomes would be that interrupting the animation process may prevent students from immersing themselves deeply in the visualization process. Comprehending an algorithm assumes that learners are able to construct an overall picture (strategy of the algorithm) from small elements (individual steps of the algorithm). Fragmenting the visualization could obstruct students in this building process. Boucheix et al. (2013) came to a similar conclusion regarding visual cueing. Their study demonstrated that cueing could counteract the spontaneous exploration of the animation.

Another contributing factor to this result could be that the increase in student engagement generated an increase in usability difficulty too. As mentioned above, although we focused on the cognitive functionality aspects, the conditions differed from the perspective of functional interactivity too. We tried to diminish this influence by including a 15-minute presentation, where it was demonstrated how animation works, especially in the interactive conditions. This aspect was also taken into account when setting the amount of time allocated to this phase of the experiment.

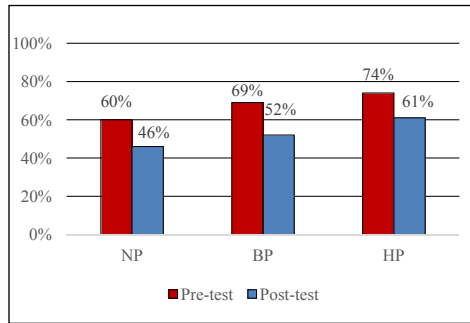
In addition, with respect to the interactive prediction method, Jarc et al. (2000) noticed that sometimes students view the integrated questions as a guessing game, and they are not taking it seriously. And, as a consequence, the objective of engaging students' attention is not reached.

12.4.1 Results grouped by prior programming experience

We also analysed participants' pre- and post-test performances grouped according to their prior programming experience (see *Figure 12.4*; independent variable: NP, BP, or HP; dependent variable: pre-test/post-test score). In both cases, as we expected, the ANOVA test showed significant differences (pre-test: $F(2,130) = 3.455$, $p = 0.034 < 0.05$; post-test: $F(2,130) = 11.299$, $p = 0.00 < 0.05$). After performing further analysis (planned contrasts), we observed that (< denotes not significant increase, << denotes marginally significant increase, and <<< denotes significant increase):

- pre-test: NP << BP < HP,
- post-test: NP < BP <<< HP.

Interestingly, while the differences between NP and HP students were almost equal in both cases (pre-test: 60% vs. 74%, post-test: 46% vs. 61%), BP students' pre-test performance was closer to their HP colleagues' performance and their post-test performance closer to NP students' performance. A possible reason could be that 66% of the post-test questions (Q1–8) focused on the operation sequence the algorithm generates, while 33% of them (Q9–12) related to algorithm efficiency issues, a concept only included in the HP students' high school curriculum.



Source: Osztián, Kátai, & Osztián, 2020

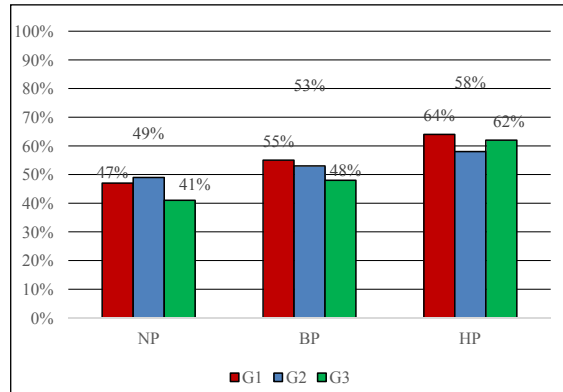
Figure 12.4. Pre- and post-test results based on prior programming knowledge

In order to investigate whether the different categories of participants (NP, BP, HP) performed equally well at each interactivity level, a two-way ANOVA was conducted. The two independent variables were the instructional condition (no, half, full interactivity) and students' prior programming experience level (NP, BP, HP), and the dependent variable was students' post-test score (Levene's test showed that the variances were equal: $p = 1.54 > 0.05$). No interaction was detected ($p = 0.6 > 0.05$). When we applied three distinct ANOVA tests for each category of students (NP, BP, HP), we also found no significant differences (see Figure 12.5).

This result is consistent with some previous research. For example, Myller et al. (2007) divided their subjects into two groups based on their pre-test results: NPK (with no prior knowledge in the topic), SPK (with prior knowledge in the topic). Although these authors emphasize that NPK students benefited more from using the visualization on a higher engagement level, they admit that the differences between the two groups are not statistically significant.

On the other hand, according to the task-appropriateness principle (Plass et al., 2009), interactive dynamic visualizations may reduce cognitive load in tasks that require high mental effort. However, they may also inhibit processing by providing unnecessary support (Schnotz & Rasch, 2005). Since (1) at

the beginning of the learning phase all participants watched the animation in viewing mode and (2) participants with prior knowledge in programming had already studied the insertion sort algorithm before (the shell sort strategy is based on this algorithm), these factors could also contribute to this result.



Source: Osztián, Kátai, & Osztián, 2020

Figure 12.5. *Post-test results by prior programming knowledge and the level of interactivity*

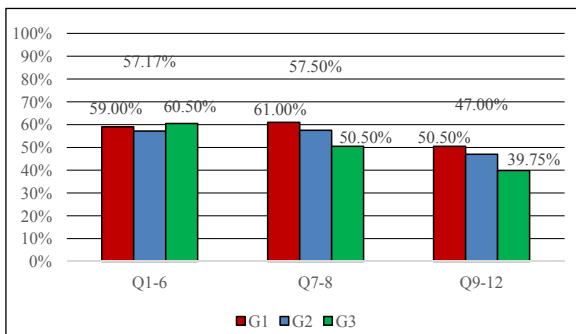
12.4.2 Relations between the level of interactivity and the nature of acquired knowledge

We designed the post-test question sequence in such a way as to have three different types of items. Questions 1–6 related to a sequence of consecutive operations (first three operations). We assumed that these kinds of questions would be particularly accessible to participants assigned to the full interactivity condition (they had to reconstruct the entire algorithm step by step: select the right pair of elements and apply the correct operation again and again). Questions 7–8 focused on sporadically selected operations from the step sequence of the algorithm. These types of questions harmonized mostly with the half interactivity condition (interactive prediction). Questions 9–12 related to algorithm complexity issues and assumed that students had managed to put together an overall picture of the algorithm. We believed that the smooth observation (no interactivity condition) of the visualization would rather contribute to this overall picture.

Students' overall performance decreased as they advanced with the post-test question sequence (Q1–6: 59%; Q7–8: 56%; Q9–12: 45%). While Q1–6 and Q7–8 scores were close to each other, the Q9–12 score was significantly lower

than these ones. Since questions 9–12 addressed algorithm complexity issues, this is a quite evident result. This is a plausible result from the perspective of Bloom’s Taxonomy of Cognitive Development too (Krathwohl, 2002). While questions 1–8 belonged to the application level, questions 9–12 represented the analysis level.

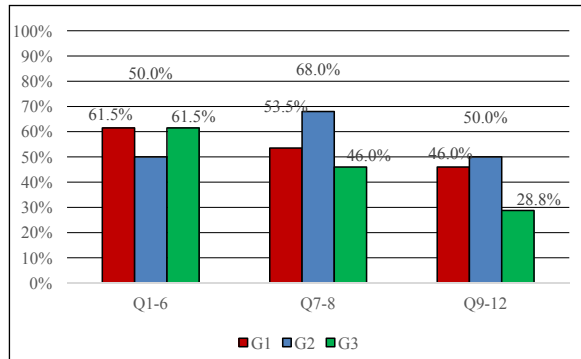
Examining the groups separately (see *Figure 12.6*), the only (marginally) significant difference was found when we compared the Q9–12 score of group G3 with the corresponding scores of groups G1 and G2 (ANOVA, contrast values $(-1, -1, 2)$; $p = 0.058$). A possible reason why students from group G3 scored significantly lower on the Q9–12 items than their colleagues from the other two groups might be that in the full interactivity condition students faced the algorithm in a very fragmented way.



Source: Osztián, Kátai, & Osztián, 2020

Figure 12.6. Results by question type and level of interactivity

As a next step, we repeated the above analysis for each category of students (NP, BP, HP). The only significant result was found with respect to BP students. When we examined the groups (G1, G2, G3) separately, we noticed interesting differences (see *Figure 12.7*). Although in the case of both group G1 and group G3 the performances decreased linearly, the slope of decrease differed (G1: 61.5%, 53.5%, 46%; G3: 61.5%, 46%, 28.8%). In addition, participants from group G2 (half interactivity; interactive prediction) performed best on questions 7–8, and they reached the highest scores on these questions. These results suggest a correlation between the level of interactivity (at which the student studies the algorithm) and the nature of acquired knowledge.

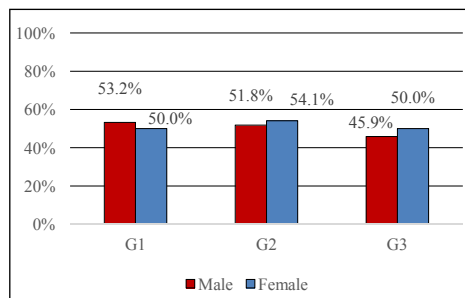


Source: Osztián, Kátai, & Osztián, 2020

Figure 12.7. Results by question type and level of interactivity – BP students

12.4.3 Results grouped by gender

Since the most effective method for male and female participants could differ, we also analysed students' post-test performances grouped by gender (see *Figure 12.8*). We found no significant differences. In order to investigate whether the gender groups performed equally well at each interactivity level, a two-way ANOVA was conducted. The two independent variables were the instructional condition (no, half, full interactivity) and gender (male, female), and the dependent variable was students' post-test score (Levene's test showed that the variances were equal: $p = 0.09 > 0.05$). No interaction was detected ($p = 0.8 > 0.05$). Despite this fact, we observed that while males performed better than females in the no interactivity condition, this relationship was reversed in the other two conditions (half and full interactivity).

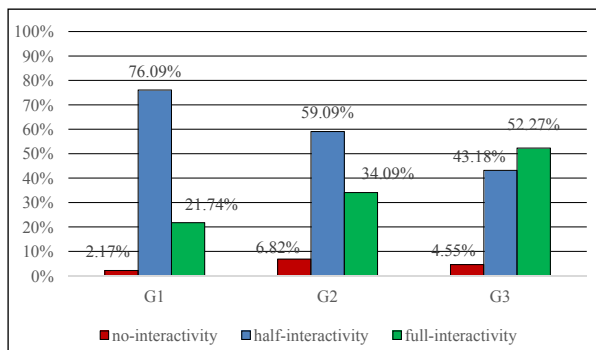


Source: Osztián, Kátai, & Osztián, 2020

Figure 12.8. Post-test results grouped by gender and interactivity level

12.4.4 Most preferred course variant

After the post-test, an extra question asked participants to indicate their most preferred interactivity level (the most preferred course variant, regardless of which they were assigned to groups). As can be discerned in *Figure 12.9*, in the case of group G1, 76.09% of the students wanted one level higher (G2) and 21.74% two levels higher (G3) interactivity. Likewise, 34.09% of the students from group G2 wanted one level higher (G3) interactivity and 59.09% were pleased with the assigned course (G2), which was associated with half interactivity. In the case of group G3, the majority of students (52.27%) were pleased with this course.



Source: Osztián, Kátai, & Osztián, 2020

Figure 12.9. Students' satisfaction level diagram

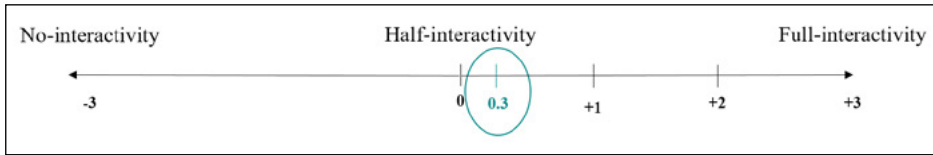
We encoded participants' choices as follows (see *Table 12.2* and *Figure 12.10*):

- 0: for those participants who were pleased with the course assigned to them,
- –1 or –2: for those who would prefer less interactivity (with one or two levels lower),
- +1 or +2: for those who would prefer more interactivity (with one or two levels higher).

Table 12.2. Students' satisfaction level

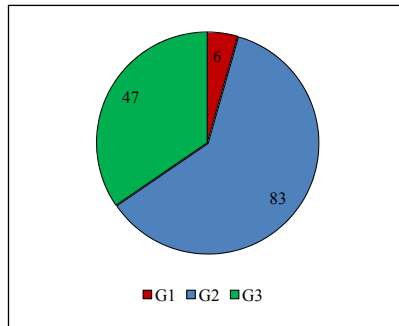
Assigned course	–2	–1	0	+1	+2	Avg
G ₁	-	-	1 (G ₁)	35 (G ₂)	10 (G ₃)	1.19
G ₂	-	3 (G ₁)	26 (G ₂)	15 (G ₃)	-	0.27
G ₃	2 (G ₁)	19 (G ₂)	23 (G ₃)	-	-	–0.52
Overall average						0.31

Source: Osztián, Kátai, & Osztián, 2020



Source: Osztián, Kátai, & Osztián, 2020

Figure 12.10. *Students' satisfaction level axis*



Source: Osztián, Kátai, & Osztián, 2020

12.11. *Most preferred course variant*

Analysis of the students' responses revealed a 0.31 mean value. This positive value suggests that participants voted for more interactivity. As shown in *Figure 12.11*, half-interactivity engagement level is the closest to students' preferences. This is encouraging since several prior studies concluded in favour of this type of engagement. In an experiment implemented by researchers Byrne, Catrambone, and Stasko (1999), video was used as a material of the learning phase. The video stopped at certain predetermined key moments, and students had to indicate what the next step would be. Grissom et al. (2003) also report that students who responded to questions integrated into the AV tool during their exploration of an algorithm showed the most improvement between a pre-test and post-test.

Interestingly, just as Naps et al. (2002) reported a strong perception among educators that visualization can help (although previous research results did not support this perception), we detected a similar phenomenon with respect to learners: they voted for more interactivity despite the fact that their performance did not provide evidence in this sense.

On the other hand, we should emphasize that students' satisfaction choice might be based on their imagination and not on real experience. In future research, we plan to reanalyse this apparently contradictory result (usually, if

students are interested in an approach, they will be more likely to be involved more actively, which results in more effective learning).

12.4.5 Limitations

One of the limitations of this study is that we included in our investigation only one algorithm: the AlgoRythmics animation of the shell sort algorithm.

Another limitation could be that participants were assigned to a group. A further research issue could be as follows: what if students could freely choose the group and control their learning (while the system records interactive steps)?

With respect to the fact that differences in the level of engagement could generate as a side-effect differences in usability factors, future investigations would test this possible influence, for example, by an extra post-test questionnaire.

12.5 Conclusions

This study provides a new insight into the research field of AV and valuable guidelines for instructional design. Based on the engagement taxonomy introduced by Naps et al. (2002), we examined three interactivity levels: viewing, responding, and constructing. We implemented these levels of engagement in the AlgoRythmics environment (using the animation that illustrates the shell sort algorithm). We identified the designed instructional conditions as: no interactivity (passive viewing), half interactivity (interactive prediction), and full interactivity (algorithm orchestration). One of the most important conclusions of this investigation is that a universally optimal interactivity level cannot be established.

Based on this study, we cannot confirm the hypothesis we formulated regarding the AlgoRythmics visualization of the shell sort algorithm – that the level of engagement would have a significant effect on the learning outcomes. Our findings suggest that all three levels of engagement that we analysed may have both advantages and disadvantages. Interestingly, Urquiza-Fuentes and Velázquez-Iturbide (2009), after examining several previous research articles in the field of AV effectiveness, also concluded that improvements in knowledge acquisition had been detected at all engagement levels.

We also found that students with different prior experience can equally benefit from each engagement level. Additionally, some of our results suggest a correlation between the level of interactivity and the nature of acquired knowledge.

Since we analysed only one specific algorithm visualization, it is clear that in order to generalize these findings further research is needed that also

addresses such possible influencing factors as algorithm complexity, usability issues, design parameter, the learning environment, etc.

Everybody deserves to learn, but everybody is different – the most effective learning style should differ in the case of each student. Therefore, we can declare that all learning environments should be able to present AV with different interactivity levels. If this can be achieved, all participants would find the best learning style for themselves, and this could only lead to better results.

13 ONGOING RESEARCH

The impetus generated by the redesign of the environment has launched further lines of research. In this penultimate chapter, we briefly present four of our running research projects.

13.1 Schematic versus human-movement-enriched realistic algorithm visualization

Two of our ongoing studies focus on the topic of schematic versus realistic dynamic visualization.

Instructional dynamic visualizations can be classified as schematic or realistic representations (Nugteren, Tabbers, Scheiter, & Paas, 2014). During the last decades, some relevant studies (from different fields of education) have focused on the relative value of these types of visualizations (Scheiter, Gerjets, Huk, Imhof, & Kammerer, 2009; Nugteren et al., 2014). However, the topic of algorithm visualization has been poorly researched from this perspective. A possible reason could be that computer algorithms are inherently abstract entities, which lack any tangible real-world representation. Consequently, schematic animations are commonly used for illustrating computer algorithms. On the other hand, there are representations, especially in the context of unplugged CS education, that are closer to realistic visualizations. A relevant example in this sense is the AlgoRythmics environment, which includes ten videos (besides schematic computer animations) illustrating basic computer algorithms by realistic dance choreographies.

Prior studies that compared the effectiveness of these two types of visualizations conclude that both types could have advantages depending on the learning outcomes that need to be accomplished (Nugteren et al., 2014). For example, based on the coherence principle of multimedia learning, a schematic animation may support learners in focusing on the important aspects of the visualization because it contains less irrelevant/distracting elements (Mayer & Moreno, 2003; Butcher, 2006). On the other hand, according to Goldstone and Son (2005), two benefits of realistic visualizations that might also be extended to the AlgoRythmics videos are: (1) they can be more easily remembered (being more concrete); (2) they promote intrinsic motivation (could be more appealing).

Surprisingly, the studies that researched whether dynamic visualizations aids learners' understanding of dynamic phenomena have reported mixed results (Ainsworth & VanLabeke, 2004). The most commonly cited reason why

animations are not found to be consistently effective is their transient nature (Stenning, 1998; Ayres & Paas, 2007). However, interestingly, dynamic visualizations have been found to be consistently superior to static visualizations when learning with animation involved human movement (Castro-Alonso, Ayres, & Paas, 2014). In this special case, it seems that the mirror neuron system assists working memory in coping with transitory information.

Accordingly, a particular strength of the AlgoRhythmic videos is that they illustrate the basic operations of the algorithms by human movements. Recent research results emphasize that observing human movements can be cognitively beneficial (Castro-Alonso, Ayres, Wong, & Paas, 2018). According to Van Gog et al. (2009), this so-called “human movement effect” (HME) (Paas & Sweller, 2012) enhances dynamic visualizations by counteracting the drawback resulted from their transient nature.

The majority of the prior studies in the field of realistic versus schematic representations report supporting evidence in the favour of schematic visualization. For example, Scheiter et al. (2009) analysed learning settings that combined realistic and schematic dynamic visualizations in the context of biology education. Results showed that participants from the realistic + realistic (the same visualization presented in succession) condition performed significantly worse than their colleagues from the other three conditions (schematic + schematic, schematic + realistic, realistic + schematic). More recently, the authors of a study (Nugteren et al., 2014) report similar results. They investigated the following learning settings: schematic-only, realistic-only, sequential schematic + realistic, simultaneous schematic + realistic. Again, the realistic-only condition scored significantly lower than the other three conditions. These findings are in accordance with the statement Tversky, Morrison, and Betrancourt (2002) made that animations should contain minimal realism because even appealing realistic details may obstruct the comprehension of the relevant movements of dynamic visualizations.

But what if the realistic animation includes visualization of human movements (HME-realistic)? Can they benefit from the HME to the point that they overcome (or be at least equally as effective as) their schematic counterparts? We initiated two investigations that focus on this topic in the context of the AlgoRhythmic learning environment.

In a previous research that investigated AlgoRhythmic videos (performed mainly by the authors of the environment), the video preceded the schematic animation of the studied algorithm mostly for motivational reasons. These studies emphasized the motivational role of the decorative elements: to arouse curiosity by providing novelty, incongruity, and surprise (Káta, 2015, 2020). In the studies in question, we focus on comparing AlgoRhythmic animations and videos as equivalent tools for helping students understand the strategies the illustrated algorithms apply.

13.1.1 Abstract animation versus dance choreography (Study 9)

Höffler and Leutner (2007) in their meta-analysis in the field of “instructional animation versus static pictures” make a distinction between representational animations (the topic to be learned is explicitly depicted in the animation) and decorative animations (the primary instructional function of the animation is to motivate the learner) (Carney & Levin, 2002). They found that animations are specifically superior to static pictures when the visualization plays a representational role (the depicted motion in the animation explicitly refers to the topic to be learned).

Based on this terminology, we identified two types of realistic/artistic elements in the AlgoRythmics videos: (1) decorative elements (specific costumes, music, male/female dancers, etc.) and (2) representational elements (specific dance steps illustrating the key operations of the algorithms). In accordance with this, we classified the potentially distracting elements as well into two categories: (1) decorative distractors and (2) representational distractors. All decorative elements could be decorative distractors. We are talking about representational distractors when realistic/artistic elements shadow the link (the one-to-one relationship) between the key operations of the algorithm and the corresponding dance steps. We were particularly interested in the influence these distracting elements may have on the effectiveness of the AlgoRythmics videos.

Two algorithms were selected for this analysis: bubble sort and selection sort. Both algorithms have $O(n^2)$ worst case time complexity. The corresponding videos illustrate the two algorithms by Hungarian (“Csángó”) and Gypsy folk dances respectively (*figures 2 and 3*). Each choreography contains several artistically enhanced decorative elements.

Both algorithms can be perceived as a succession of comparing and comparing + swapping operations. During the selection sort video, the comparison operations are illustrated consistently by the same dance steps (whether or not they are combined with a swapping operation). On the other hand, in the case of the bubble sort video, the combined comparing + swapping scenes do not include clearly separable comparing and swapping phases. Because of this particularity, we considered that the bubble sort choreography contains not only decorative but also representational distractors.

The two instructional conditions we implemented are bubble sort + selection sort (presented in succession, one after the other) illustrated by (1) schematic animations or (2) HME-enhanced realistic videos. In line with the above-detailed prior research, we anticipated the following:

- Students assigned to the “realistic condition” will not perform less well than their counterparts from the “schematic condition group” (for example, because of the HME).

- For the members of the “realistic condition group”, it will be easier to distinguish between the two algorithms (since as realistic visualizations they can be more easily remembered).
- It is not so much the decorative elements as the representational distractors that can obstruct the comprehension of the relevant movements of the realistic algorithm visualizations.

13.1.2 Combining schematic and realistic visualizations in the AlgoRhythmic environment (Study 10)

Based on the complementary roles of multiple external representations (MER), some research analysed learning environments that combine realistic and schematic dynamic visualizations. According to Van Gendt and Verhagen (2001), for example, realistic visualizations may support the identification of structures and processes in the studied real systems, while schematic animations may strengthen the comprehension of underlying principles. Additionally, mostly because of the constraining function of MERs, there are studies with particular focus on how to combine these visualizations temporally.

There are a few studies that report research results regarding the relative value of realistic and schematic visualizations when these are combined as MERs of the same system, concept, or process. In a research (Moreno, Ozogul, & Reisslein, 2011), for some participants, the studied diagrams (static visualizations) were either realistic or schematic, while for others both realistic and schematic diagrams were presented simultaneously (they analysed them in realistic-schematic order). Results showed that the “schematic-only group” outperformed the “realistic-only group”, and the group that received both diagrams performed better than those that received only one representation. Two above-referred studies analysed learning settings that combined realistic and schematic dynamic visualizations. Scheiter et al.’s (2009) study concludes that participants from the realistic + realistic condition performed significantly worse than their colleagues from the other three conditions (schematic + schematic, schematic + realistic, realistic + schematic). The authors of another study (Nugteren et al., 2014) also report that the realistic-only condition scored significantly lower than the other three conditions (schematic-only, sequential schematic + realistic, simultaneous schematic + realistic).

In our second study on the “HME-realistic versus schematic” topic (multiple dynamic representations that combine human-movement-effect-enriched realistic video illustrations and schematic animations), we proposed to investigate the relative effectiveness of the following learning settings: schematic + schematic, HME-realistic + HME-realistic, schematic + HME-realistic, and HME-realistic + schematic (presented in succession). We used as HME-realistic visualization the AlgoRhythmic video of the shell sort algorithm (see *Chapter 7*).

For this investigation, four hypotheses were derived from the related theories and the above-detailed empirical insights that prior research in the field has provided. Although previous studies, such as Scheiter et al. (2009), Moreno et al. (2011), or Nugteren et al. (2014), concluded that realistic-only conditions might be inferior to schematic-only or combined conditions, we assumed that, due to HME, conditions that involve human movement would score better than HME-free conditions (Hypothesis 1). This would be in line with those studies that revealed specific benefits of HME on learning with animations (de Koning & Tabbers, 2011; Castro-Alonso et al., 2018).

Based on the literature on learning with MERs (Ainsworth, 2006; Rau et al., 2015), we hypothesized that combined visualizations would support learning better than repeatedly viewing the same visualization (Hypothesis 2). In particular, we assumed that watching the schematic visualization first would help interpreting the realistic one (cf. constraining function of MERs – Ainsworth, 2006) and, consequently, would contribute to better learning outcomes than the realistic + schematic condition (Hypothesis 3).

The above hypotheses suggest the following order: schematic + HME-realistic, HME-realistic + schematic, HME-realistic twice, schematic twice. On the other hand, due to the abundance of potentially distracting decorative elements incorporated in the Algorhythmics dance choreographies (boys, girls, traditional costumes, captivating dance steps, vivid music, etc.), it seemed likely that in order to benefit maximally from the HME-realistic visualization students will need either a second viewing or a previous schematic presentation. Accordingly, we hypothesized that the realistic-last conditions will outperform the schematic-last ones (Hypothesis 4). This outcome would also be in line with the assumption that deeper understanding will occur during the second viewing (Ainsworth, 2006), and realistic visualization can be more easily remembered (Goldstone & Son, 2005).

13.2 Improving AlgoRhythmics teaching-learning environment by asking questions (Study 11)

A possible approach to making computing education attractive for different categories of learners (including K–12 learners and non-CS majors) is contextualization (Guzdial, 2010). For example, in the case of non-CS majors, the context should be related to the major field of the students. Since developing differentiated teaching-learning strategies may involve substantial additional costs, some scholars have tried to find a context that is appealing to most students. A promising candidate for this “common denominator role” could be arts (Tew, McCracken, & Guzdial, 2005; Guzdial & Tew, 2006; Simon et al., 2010; Daily et al., 2014; Wood, Muhl, & Hicks, 2016).

The AlgoRythmics learning environment was designed along this approach. Since music and dance are relatively close to most young people, this environment visualizes basic computer algorithms (searching and sorting) by professional dance choreographies (folkdance, flamenco, ballet). Previous research on the AlgoRythmics environment concentrates mostly on the potential the dance choreographies (supplemented with animations) incorporate to support different categories of students in understanding the strategy of the algorithms (Kátai, 2014a–c, 2015, 2020). Other characteristics of these studies are that the authors analyse only self-paced learning sessions, and the examined learning environments supplement the viewing of the dance choreographies (and the attached animations) with the interactive orchestration of the algorithms. For example, Kátai's (2015, 2020) studies (see *Chapter 10*) focus on sciences- versus humanities-oriented undergraduate students and conclude that active involvement plays a crucial role in supporting humanities-oriented students in assimilating the strategy of the studied algorithm. In addition, the above-referred studies do not investigate if students are able to build on the knowledge they have acquired.

In this study, we analyse a learning setting built around AlgoRythmics videos where the principle of active involvement is implemented by asking questions (with and without teacher guidance). Besides prior research in the field, our experience with these videos (as CS teachers) also inspired us since we have observed that without teacher support the potential these choreographies incorporate as CT promoter tools is only partially exploited. For example, definitions often present the concept of algorithm efficiency as an important component of CT (Shute et al., 2017; CSTE, 2017), and this concept is based on the best- and worst-case behaviour of algorithms. We have proposed to investigate the following question: are these visualizations (created for specific inputs) expressive enough to support students without prior knowledge in computing to imagine the best- and worst-case behaviour of algorithms? The attached questions, besides supporting and guiding students' thinking process, also help them focus on the domain-relevant aspects of the visualizations.

13.3 Investigating young school students' computational thinking ability across grade levels (Study 12)

Over the last decade, continuous efforts have been made to bring CT closer to K–12 education and even to K–9 education. Two complementary implementation approaches for the “CT for all” initiative are: (1) introducing new computing courses and (2) infusing CT into existing ones (Román-González, Pérez-González, Moreno-León, & Robles, 2018). For example, since 2014, computer science has

been mandatory for pupils in the UK from age five upwards (Brown, Sentance, Crick, & Humphreys, 2014), and a growing recognition of the importance of CS education is observable in other countries too (European Schoolnet, 2015). In addition, Mannila et al. (2014) report on a survey distributed to K–9 teachers, aiming at revealing to what extent different aspects of CT are already part of the current curricula in various European countries and the U.S.

These focused endeavours implicitly suggest that current curricula do not contribute sufficiently to the development of learners' CT. Several studies emphasize that computing students lack a variety of skills that programming ability would require (Ahadi, Lister, Lal, Leinonen, & Hellas, 2017; Evans & Simkin, 1989; Simon, Chen, Lewandowski, McCartney, & Sanders, 2006). On the other hand, since CT is a combined skill with cross-disciplinary implications (Feaster, Ali, Zhai, & Hallstrom, 2014), one might conclude that, even without an explicit focus on CS education, students' CT might develop latently as they advance with the current curriculum. For example, Lewandowski and his colleagues report on a series of computing projects with the goal of identifying the commonsense knowledge beginners bring to the study of CS (Lewandowski et al., 2010).

While the majority of previous research focuses on assessing the CT level of certain age-groups, the present study was motivated by the following basic question: Is there any detectable incidental progress in students' CT during their K–9 education? Instead of using one of the general CT tests proposed by previous works in the field (Román-González et al., 2018; Shute et al., 2017), we chose to follow the same approach the authors of the above-referred computing project applied. More explicitly, the main research question we addressed is: are there any detectable differences in how 3rd-, 5th-, 7th-, and 9th-grade learners (without any explicit prior experience with CT) relate to learning tasks that assume a certain level of CT? More specifically:

- What is the pace of the potential CT growth?
- Does the rate and pace of CT growth depend on the nature of the current curriculum (arts vs. theoretical schools)?
- To what extent can students of different grade levels assimilate a basic computer algorithm (linear search)?
- To what extent are there signs of advanced CT at different grade levels?
- Does the rate and pace of CT growth depend on the gender of the learners?

14 ALGORHYTHMICS: PAST, PRESENT, AND FUTURE

The AlgoRhythmic project started more than fifteen years ago. The main reason behind this initiative was to build a multi-sensorial platform for promoting algorithmic and computational thinking. In 2013, the project was awarded the *Best Practices in Education Award* (Informatics Europe, 2013). Some strengths of that version of the learning environment (also appreciated by the evaluation committee) were that it invited students on an artistically enhanced, apparently CS-free, interactive tour of representative regions of the exciting world of sorting algorithms. Another advantage of the AlgoRhythmic videos is that the algorithms are illustrated by human movement (dance choreographies). Recent research results emphasize that observing human movements (or producing body movements on our own) can be cognitively beneficial (Castro-Alonso et al., 2018). According to these authors, our cognitive systems are wired to observe human movements.

14.1 The renewed learning environment

On the other hand, the initial learning tool had some evident limitations: it included only one type of algorithms (six sorting strategies) and, more importantly, did not support students in developing the code of the studied algorithms. During the past years, we have focused on these aspects. Another priority was to increase the diversity of the visualizations with regard to dance styles and data structure representations.

We expanded the AlgoRhythmic environment along four dimensions:

- Algorithmic dimension – the original collection was extended with four new algorithms: heap sort, two searching algorithms (linear, binary), and a backtracking strategy.
- Artistic dimension – besides folk dances, new dance styles were introduced: flamenco and ballet.
- Data structure visualization – in the case of the first six algorithms, we used 1D visualizations of the number sequence to be sorted. The new heap sort visualization explicitly displays how a unidimensional array can be perceived as a binary tree.
- From dance to code – as the most important expansion, two new modules were added to the learning environment: after students have assimilated the studied algorithmic strategy, the newly developed interactive web

application continues to guide and support them in creating the code of the algorithm.

The “dance floor” we created is an interactive and intuitive user interface which helps in initiating learners into computer algorithms. The “choreographer” (teacher) can predefine courses (sequences of various learning steps) in a dynamic way which also involves the specification of the level of the user interaction. If a “dancer” (user) catches the rhythm of it, he/she will easily go through it, resulting in the acquisition of the algorithmic knowledge. In the meantime, a mirror is turned towards the user, which helps monitoring his/her personal evolution. The generated log files constitute a valuable database for educational research.

Learning steps can be considered the atomic elements. These constitute that new dimension through which we can get from the dance to the code. We have defined five basic ones.

- *Video*. The algorithms are illustrated by dance choreographies. Users can choose to simply watch the video or be actively involved in the visualization (by indicating the next operation every time when the video stops at predetermined moments).
- *Animation*. Watching the abstract animation of the algorithm might be an important step from the dance to the code. A similar interactivity is available as in the case of the previous learning step. In addition, the animations can be played for several inputs (teacher, random, best case, and worst case) stored in white or black arrays.
- *In control*. Taking the user experience to a new level, we give the control to the user. Using the already gained knowledge about a selected algorithm, the user has the possibility to manually play the whole algorithm.
- *Create code*. During this learning step, the enhanced animation of the algorithm is played three times. The first play attaches sound effects to the animation in order to help students in selecting the proper loop structure for the code (in the case of iterative implementations). The second and third plays support students in completing the partial code of the algorithm.
- *The code comes alive*. Summarizing all the previous elements, the user can pay attention to the result of his/her work. The synchronized animation is played parallel with the step-by-step execution of the code (the relevant code lines are highlighted).

Courses constitute the main concept of the learning environment. Their role is to help users understand the algorithms and to support educational research. As described above, the involved learning steps can be differentiated by their input, display, and playback type. Different combinations of learning steps of many variations create different courses. These can be defined dynamically by the administrators or teachers.

14.2 Research in the AlgoRhythmics environment

In this book, we especially focused on the research we had performed in the AlgoRhythmics environment. Findings of the studies detailed in chapters 3, 4, 5, 8, 9, 10, and 12 (Studies 1–8) were published in reputable journals and conference proceedings such as: *Computers & Education*, *Journal of Computer Assisted Learning*, *Teaching and Teacher Education*, *Educational Technology Research and Development*, *Computer Applications in Engineering Education*, *Innovation and Technology in Computer Science Education (ITiCSE)*, and *Frontiers in Education (FIE)*. The studies presented in *Chapter 13* (Studies 9–12) are submitted for publication.

14.3 Plans for the near future

We have ambitious plans for the future. We are especially interested in the following research questions:

- Which order of learning steps appears to be the most effective?
- Which is the best AlgoRhythmics recipe for non-CS students or in the case of children?

With respect to the learning environment, we are planning to design:

- parallel versions of quick and merge sort dances,
- visualizations that allow users to replace one of the dancers (by creating an avatar with Kinect sensor).

14.4 Final conclusion

The algorithmic dance choreography collection is still the core of the AlgoRhythmics environment. During the whole project, we were determined to enrich the meaning of algorithm with algo-rhythm. The resulted science–art combinations were incorporated into a flexible web application, the content of which can be dynamically changed and the level of user interaction managed as well.

From the perspective of the teaching-learning process, the most important features of the environment are its unified, artistically enhanced, human-movement-effect-enriched, multisensory, and interactive character. In addition, the generated log files are valuable sources for educational research.

Due to AlgoRhythmics, the association of dance, rhythm, and algorithm are not strange anymore. We believe we have managed to break down the stereotypes proclaiming that learning is boring. Moderate-progressive challenge plus genuine active involvement together with a little bit of art can transform learning into an exciting experience. As a famous quote states: with AlgoRhythmics, *we are bold and italic but never regular*.

REFERENCES

- Achter, J., Lubinski, D., & Benbow, C. (1999). Assessing vocational preferences among gifted adolescents adds incremental validity to abilities: A discriminant analysis of educational outcomes over a 10-year interval. *Journal of Educational Psychology, 91*(4), 777–786.
- Adesope, O. O., & Nesbit, J. C. (2012). Verbal redundancy in multimedia learning environments: A meta-analysis. *Journal of Educational Psychology, 104*(1), 250.
- Ahadi, A., Lister, R., Lal, S., Leinonen, J., & Hellas, A. (2017, January). Performance and Consistency in Learning to Program. In *Proceedings of the Nineteenth Australasian Computing Education Conference* (pp. 11–16). ACM.
- Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 832–835.
- Ainsworth, S. (2006). DeFT: A conceptual framework for considering learning with multiple representations. *Learning and instruction, 16*(3), 183–198.
- Ainsworth, S., & VanLabeke, N. (2004). Multiple forms of dynamic representation. *Learning and instruction, 14*(3), 241–255.
- Amaram, D. I. (2007). Cultural Diversity: Implications for Workplace Management. *Journal of Diversity Management, 2*(4), 1–6.
- American Council on Education, Presidents' Task Force on Teacher Education. (1999). *To touch the future: Transforming the way teachers are taught*. <http://www.acenet.edu/bookstore/pdf/teacher-ed-rpt.pdf>.
- Ausburn, L. J., & Ausburn, F. B. (1978). Cognitive styles: Some information and implications for instructional design. *Educational Communication and Technology, 26*, 337–354.
- Ayres, P., & Paas, F. (2007). Making instructional animations more effective: A cognitive load approach. *Applied Cognitive Psychology, 21*, 695–700.
- Baldwin, M., & Rivers, R. (2005). Dancing to the ideas of Einstein. *Physics World, 18*(5), 16–17.
- Banks, J. A. (1977). Pluralism and educational concepts: A clarification. *Peabody Journal of Education, 54*(2), 73–78.
- Banks, J. A. & Banks, C. A. M. (2001). *Multicultural Education: Issues and Perspectives* (4th ed.). New York: John Wiley & Sons, Inc.
- Baptiste, H. P. (1979). *Multicultural education: A synopsis*. Washington, D.C.: University Press of America.
- Barger, A., & Byrd, K. (2011). Motivation and computer-based instructional design. *Journal of Cross-Disciplinary Perspectives in Education, 4*(1), 1–9.
- Baron-Cohen, S. (2003). *The essential difference: Men, women and the extreme male brain*. London: Penguin.

- Baron-Cohen, S., Knickmeyer, R., & Belmonte, M. K. (2005). Sex Differences in the Brain: Implications for Explaining Autism. *Science*, *310*, 819–823.
- Bebras. (2020). <https://www.bebras.org/>.
- Bennett, C. I. (1990). *Comprehensive multicultural education: Theory and practice* (2nd ed.). Boston: Allyn and Bacon.
- Bennett, M. J. (1993). Towards ethnorelativism: A developmental model of intercultural sensitivity. *Education for the Intercultural Experience*, *2*, 21–71.
- Berg, D., & Knop, F. N. (2008). Making math real: Connecting research to practice – A comprehensive multisensory structured methodology in mathematics K–12, *Learning & the brain*, February 7–9, San Francisco. <http://www.edupr.com/workshopssf.html>.
- Berlyne, D. E. (1960). *Conflict, arousal, and curiosity*. New York: McGraw Hill Book Company, Inc.
- Berney, S., & Bétrancourt, M. (2016). Does animation enhance learning? A meta-analysis. *Computers & Education*, *101*, 150–167.
- Beyth-Marom, R., Saporta, K., & Caspi, A. (2005). Synchronous vs. asynchronous tutorials: Factors affecting students' preferences and choices. *Journal of Research on Technology in Education*, *37*, 245–262.
- Billington, J., Baron-Cohen, S., & Wheelwright, S. (2007). Cognitive style predicts entry into physical sciences and humanities: Questionnaire and performance tests of empathy and systemizing. *Learning and Individual Differences*, *17*, 260–268.
- Blumenfeld, P. C. (1992). Classroom learning and motivation: Clarifying and expanding goal theory. *Journal of Educational psychology*, *84*(3), 272.
- Blumenfeld, P. C., & Meece, J. L. (1988). Task factors, teacher behavior, and students' involvement and use of learning strategies in science. *The Elementary School Journal*, *88*(3), 235–250.
- Bongard, M., Ferrandez, J. M., & Fernandez, E. (2009). The neural concert of vision. *Neurocomputing*, *72*(4–6), 814–819.
- Boucheix, J. M., & Guignard, H. (2005). What animated illustrations conditions can improve technical document comprehension in young students? Format, signaling and control of the presentation. *European Journal of Psychology of Education*, *20*(4), 369–388.
- Boucheix, J. M., Lowe, R. K., Putri, D. K., & Groff, J. (2013). Cueing animations: Dynamic signaling aids information extraction and comprehension. *Learning and Instruction*, *25*, 71–84.
- Bransford, J., Brown, A., & Cocking, R. (1999). How people learn: Brain, mind, experience, and school. In: *Committee on Developments in the Science of Learning, Commission on Behavioral and Social Sciences and Education*, National Research Council (ed.), Washington, DC: National Academy Press.
- Brantingham, A. (2011). Understanding. *Macalester Journal of Philosophy*, *12*(1), 4.

- Bridges Organization. (2004). *Bridges: Mathematical connections in art, music, and science*. <http://www.bridgesmathart.org/past-conferences/2004-2>.
- Brown, N. C., Sentance, S., Crick, T., & Humphreys, S. (2014). Restart: The resurgence of computer science in UK schools. *ACM Transactions on Computing Education (TOCE)*, 14(2), 9.
- Burg, J., & Lüttringhaus, K. (2006, April/June). Entertaining with science, educating with dance. *Computers in Entertainment*, 4(2). from: <http://portal.acm.org>.
- Butcher, K. R. (2006). Learning from text with diagrams: Promoting mental model development and inference generation. *Journal of Educational Psychology*, 98(1), 182.
- Byrne, M. D., Catrambone, R., & Stasko, J. T. (1996). *Do algorithm animations aid learning?* Georgia Institute of Technology.
- Byrne, M. D., Catrambone, R., & Stasko, J. T. (1999). Evaluating animations as student aids in learning computer algorithms. *Computers & education*, 33(4), 253–278.
- Calvert, G., Spence, C., & Stein, B. E. (2004). *The handbook of multisensory processes*. Cambridge: MIT Press.
- Campbell, D. (2000). *The Mozart effect for children: Awakening your child's mind, health, and creativity with music*. New York: Harper Collins.
- Carey, S. (1985). *Conceptual change in childhood*. MIT press.
- Carney, R. N., & Levin, J. R. (2002). Pictorial illustrations still improve students' learning from text. *Educational Psychology Review*, 14(1), 5–26.
- Castro-Alonso, J. C., Ayres, P., & Paas, F. (2014). Dynamic visualisations and motor skills. In: *Handbook of human centric visualization* (pp. 551–580). Springer, New York, NY.
- Castro-Alonso, J. C., Ayres, P., Wong, M., & Paas, F. (2018). Learning symbols from permanent and transient visual presentations: Don't overlay the hand. *Computers & Education*, 116, 1–13.
- Chavey, D. (1996). Songs and the analysis of algorithms. In: *Proceedings of the 27th SIGCSE technical symposium on computer science education* (pp. 4–8).
- Christophel, D. M. (1990). The relationships among teacher immediacy behaviors, student motivation, and learning. *Communication Education*, 39, 323–340.
- Clark, R. C., & Mayer, R. E. (2002). *E-learning and the science of instruction*. San Francisco: Jossey-Bass/Pfeiffer.
- Coates, D., Humphreys, B. R., Kane, J., & Vachris, M. A. (2004). No significant distance between face-to-face and online instruction: Evidence from principles of economics. *Economics of Education Review*, 23, 533–546.
- Computer Science Teachers Association (CSTA). (2020). <https://www.csteachers.org/>.

- Council of Europe, Committee on Culture, Science and Education. (2009). *Cultural education: The promotion of cultural knowledge, creativity and intercultural understanding through education*. <http://assembly.coe.int>.
- Csikszentmihályi, M. (1990). *Flow: The psychology of optimal experience*. New York: Harper Perennial.
- Daily, S. B., Leonard, A. E., Jörg, S., Babu, S., & Gundersen, K. (2014, March). Dancing Alice: Exploring embodied pedagogical strategies for learning computational thinking. In: *Proceedings of the 45th ACM technical symposium on Computer science education* (pp. 91–96). ACM.
- Dalcrose Society of America (DSA). (2020). <http://www.dalcrozeusa.org>.
- Dani, E. (2016). The HY-DE Model: An interdisciplinary attempt to deal with the phenomenon of hyperattention. *Journal of Systemics, Cybernetics and Informatics*, 13(6), 8-14.
- Davies, A., Fidler, D., & Gorbis, M. (2011). *Future work skills 2020*. Institute for the Future for University of Phoenix Research Institute, 540.
- de Koning, B. B., & Tabbers, H. K. (2011). Facilitating understanding of movements in dynamic visualizations: An embodied perspective. *Educational Psychology Review*, 23(4), 501–521.
- Deci, E. L., & Ryan, R. M. (1985). *Intrinsic motivation and self-determination in human behavior*. New York: Plenum.
- Dede, C., Salzman, M. C., Loftin, R. B., & Sprague, D. (1999). In: Nancy Roberts, Wallace Feurzeig, & Beverly Hunter (eds.), *Multisensory immersion as a modeling environment for learning complex scientific concepts, computer modeling and simulation in science education*. Springer-Verlag.
- Denning, P. J. (2009). The profession of IT Beyond computational thinking. *Communications of the ACM*, 52(6), 28–30.
- Denning, P. J. (2017). Remaining trouble spots with computational thinking. *Communications of the ACM*, 60(6), 33–39.
- Dobbie, A. M. O. (1986). *Comenius's Pampaedia or Universal education*. Buckland: Dover.
- Douadi, B., Tahar, B., & Hamid, S. (2012). Smart edutainment game for algorithmic thinking. *Procedia – Social and Behavioral Sciences*, 31, 454–458.
- Driver, J., & Noesselt, T. (2008). Multisensory interplay reveals crossmodal influences on ‘sensory-specific’ brain regions, neural responses, and judgments. *Neuron*, 57, 11–23.
- Driver, R. (1989). Students’ conceptions and the learning of science. *International Journal of Science Education*, 11(5), 481–490.
- du Boulay, B. (1986). Some difficulties of learning to program. *Journal of Educational Computing Research*, 2(1).
- du Boulay, B., O’Shea, T., & Monk, J. (1981). The black box inside the glass box: presenting computing concepts to novices. *International Journal of Man-Machine Studies*, 14(3), 237–249.

- Durndell, A., & Haag, Z. (2002). Computer self efficacy, computer anxiety, attitudes towards the Internet and reported experience with the Internet, by gender, in an East European sample. *Computers in human behavior*, 18(5), 521–535.
- Eckerdal, A., McCartney, R., Moström, J. E., Ratcliffe, & M., Zander, C. (2006). Can graduating students design software systems? In *SIGCSE'06: Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education*. New York: ACM Press.
- Eisenhower SCIMAST. (1997). Research on the brain. *Classroom Compass*, 3(2). <http://www.sedl.org>
- Eisenhower SCIMAST. (1998). Rhythm of Mathematics. *Classroom Compass*, 4(2). <http://www.sedl.org>.
- European Commission, Education, Audiovisual and Culture Executive Agency. (2009). *Arts and Cultural Education at School in Europe*. EACEA P9 Eurydice.
- European Schoolnet. (2015). *Computing our future. Computer programming and coding: Priorities, school and initiatives across Europe [Technical report]*. <http://www.eun.org/resources/detail?publicationID=661>.
- Evans, C., & Gibbons, N. J. (2007). The interactivity effect in multimedia learning. *Computers & Education*, 49(4), 1147–1160.
- Evans, G. E., & Simkin, M. G. (1989). What best predicts computer proficiency? *Communications of the ACM*, 32(11), 1322–1327.
- Fair, E. M., & Silvestri, L. (1992). Effects of rewards, competition and outcome on intrinsic motivation. *Journal of Instructional Psychology*, 19(1), 3.
- Falchier, A., Clavagnier, S., Barone, P., & Kennedy, H. (2002). Anatomical evidence of multimodal integration in primate striate cortex. *Journal of Neuroscience*, 22, 5749–5759.
- Fassbender, E., Richards, D., Bilgin, A., Thompson, W. F., & Heiden, W. (2012). VirSchool: The effect of background music and immersive display systems on memory for facts learned in an educational virtual environment. *Computers & Education*, 58(1), 490–500.
- Feaster, Y., Ali, F., Zhai, J., & Hallstrom, J. O. (2014, June). Serious toys: Three years of teaching computer science concepts in K–12 classrooms. In: *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education* (pp. 69–74). ACM.
- Field, J. (2010). Middle school music curricula and the fostering of intercultural awareness. *Journal of Research in International Education*, 9(1), 5–23.
- Filpisan, M., Tomuletiu, A. E., Gyorgy, M., & Moldovan, T. (2012). Practical guide of intercultural education. *Procedia – Social and Behavioral Sciences*, 46, 5523–5528.
- Finn, A., & Bucciari, M. (2004). *A case study approach to blended learning*. Los Angeles: Centra Software.

- Finneran, C. M., & Zhang, P. (2005). Flow in computer-mediated environments: Promises and challenges. *Communications of the association for information systems*, 15(1), 4.
- Fishwick, P. A., Diehl, S., Prophet, J., & Lowgren, J. (2005). Perspectives in aesthetic computing. *Leonardo*, 38(2), 133–141.
- Fjermestad, J., Hiltz, S. R., & Zhang, Y. (2005). Effectiveness for students: Comparisons of in-seat and ALN courses. In: S. R. Hiltz, & R. Goldman (eds.), *Learning together online: Research on asynchronous learning networks* (pp. 39–80). Mahwah, NJ: Erlbaum.
- Foxe, J. J., & Schroeder, C. E. (2005). The case for feedforward multisensory convergence during early cortical processing. *NeuroReport*, 16, 419–423.
- Frazier, L. (1977). The multicultural facet of education. *Journal of Research and Development in Education*, 11, 10–16.
- Friedl, R., Höppler, H., Ecard, K., Scholz, W., Hannekum, A., Oechsner, W., & Stracke, S. (2006). Comparative evaluation of multimedia driven, interactive, and case-based teaching in heart surgery. *The Annals of Thoracic Surgery*, 82(5), 1790–1795.
- Futschek, G. (2006, November). Algorithmic thinking: The key for understanding computer science. In: *International Conference on Informatics in Secondary Schools – Evolution and Perspectives* (pp. 159–168). Springer, Berlin, Heidelberg.
- Futschek, G. (2007). Logo-like learning of basic concepts of algorithms-having fun with algorithms. In *Proceedings of the 11th European Logo Conference* (Ed Kalas I.). Bratislava. http://publik.tuwien.ac.at/files/pub-inf_4696.pdf.
- Futschek, G., & Moschitz, J. (2010). Developing algorithmic thinking by inventing and playing algorithms. *Proceedings of the 2010 Constructionist Approaches to Creative Learning, Thinking and Education: Lessons for the 21st Century (Constructionism 2010)*, 1–10.
- Gadsden, V. L. (2008). The arts and education: Knowledge generation, pedagogy, and the discourse of learning. *Review of Research in Education*, 32(1), 29–61.
- Gardner, H. (1993). *Frames of mind* (The tenth anniversary edition). New York: Basic Books.
- Garris, R., Ahlers, R., & Driskell, J. E. (2002). Games, motivation, and learning: A research and practice model. *Simulation & gaming*, 33(4), 441–467.
- Gay, G. (1994). *A Synthesis of scholarship in multicultural education*. North Central Regional Educational Laboratory. <http://www.ncrel.org/sdrs/areas/issues/educatrs/leadrshp/le0gay.htm>.
- Gay, G. (2000). *Culturally Responsive Teaching: Theory, Research, and Practice*. New York: Teachers College Press.
- Ghazanfar, A. A., & Schroeder, C. E. (2006). Is neocortex essentially multisensory? *Trends in Cognitive Sciences*, 10, 278–285.

- Goldstone, R. L., & Son, J. Y. (2005). The transfer of scientific principles using concrete and idealized simulations. *The Journal of the Learning Sciences, 14*(1), 69–110.
- Grant, C. A. (1977). *Multicultural education: Commitments, issues, and applications*. Washington, D.C.: Association for Supervision and Curriculum Development.
- Gray, S., Edwards, S., Lewandowski, G., & Shende, A. (2005). Improving student programming skills by developing program comprehension abilities: Panel discussion. *Journal of Computing Sciences in Colleges, 20*(3), 235–237.
- Grigorovici, D., Nam, S., & Russill, C. (2003). The effects of online syllabus interactivity on students' perception of the course and instructor. *The Internet and Higher Education, 6*, 41–52.
- Grissom, S., McNally, M. F., & Naps, T. (2003, June). Algorithm visualization in CS education: Comparing levels of student engagement. In: *Proceedings of the 2003 ACM Symposium on Software Visualization* (pp. 87–94).
- Grossberg, S. (2013). Adaptive resonance theory. *Scholarpedia, 8*(5), 1569. http://www.scholarpedia.org/article/Adaptive_resonance_theory.
- Grover, S., & Pea, R. (2013). Computational thinking in K–12. A review of the state of the field. *Educational Researcher, 42*(1), 38–43.
- Guzdial, M. (2008). Education paving the way for computational thinking. *Communications of the ACM, 51*(8), 25–27.
- Guzdial, M. (2010). Does contextualized computing education help? *ACM Inroads, 1*(4), 4–6.
- Guzdial, M., & Tew, A. E. (2006, September). Imagineering inauthentic legitimate peripheral participation: An instructional design approach for motivating computing education. In: *Proceedings of the Second International Workshop on Computing Education Research* (pp. 51–58). ACM.
- Hamari, J., Shernoff, D. J., Rowe, E., Coller, B., Asbell-Clarke, J., & Edwards, T. (2016). Challenging games help students learn: An empirical study on engagement, flow and immersion in game-based learning. *Computers in human behavior, 54*, 170–179.
- Hamid, A. A. (2001). e-Learning: Is it the “e” or the learning that matters? *The Internet and Higher Education, 4*(3–4), 311–316.
- Hammel, W. C. (2002). *An essay on patterns in musical composition transformations, mathematical groups, and the nature of musical substance*. <http://graham.main.nc.us/wbhammel/MUSIC/compose.html>.
- Hansen, S. R., Narayanan, N. H., & Schrimpscher, D. (2000). Helping learners visualize and comprehend algorithms. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning, 2*(1). <http://imej.wfu.edu/articles/2000/1/02/index.asp>.
- Harding, S. (1998). *Is science multicultural?* Bloomington: Indiana University Press.

- Harp, S. F., & Mayer, R. E. (1998). How seductive details do their damage: A theory of cognitive interest in science learning. *Journal of Educational Psychology, 93*, 187–198.
- Hattie, J. (2012). *Visible learning for teachers: Maximizing impact on learning*. Routledge.
- Hoffman, D. M. (1996). Culture and self in multicultural education: Reflections on discourse, text, and practice. *American Educational Research Journal, 33*(3), 545–69.
- Höffler, T. N., Leutner, D. (2007). Instructional animation versus static pictures: A meta-analysis. *Learning and instruction, 17*(6), 722–738.
- Hu, C. (2011, June). Computational thinking: What it might mean and what we might do about it. In: *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education* (pp. 223–227). ACM, New York.
- Hundhausen, C. D. (1999). *Toward effective algorithm visualization artifacts: Designing for participation and communication in an undergraduate algorithms course* (pp. 1–361). University of Oregon.
- Hundhausen, C. D., Douglas, S. A., & Stasko, J. T. (2002). A meta-study of algorithm visualization effectiveness. *Journal of Visual Languages & Computing, 13*(3), 259–290.
- Hung, C. Y., Sun, J. C. Y., & Yu, P. T. (2015). The benefits of a challenge: student motivation and flow experience in tablet-PC-game-based learning. *Interactive Learning Environments, 23*(2), 172–190.
- Hung, D. (2003). Supporting current pedagogical approaches with neuroscience research. *Journal of Interactive Learning Research, 14*(2), 129–155.
- Hunter, W. A. (Ed.). (1974). *Multicultural education through competency-based teacher education*. Washington, D.C.: American Association of Colleges for Teacher Education.
- I Programmer. (2011). *Sorting Algorithms as dances*. <https://www.i-programmer.info/news/150-training-a-education/2255-sorting-algorithms-as-dances.html>.
- Informatics Europe. (2013). <http://www.informatics-europe.org/services/curriculum-award.html>.
- Informatics Europe & ACM Europe Working Group on Informatics Education (IE & ACM). (2013). *Informatics education: Europe cannot afford to miss the boat*. <https://www.informatics-europe.org/images/documents/informatics-education-acm-ie.pdf>.
- International Society for Technology in Education (ISTE). (2020). <http://www.iste.org/>.
- Irons, L. R., Keel, R., & Bielema, C. L. (2002). Blended learning and learner satisfaction: Keys to user acceptance? *USDLA Journal, 16*(12). http://www.usd-la.org/html/journal/DEC02_Issue/article04.html.

- Jain, A. K., & Duin, R. P. W. (2004). Pattern recognition. In: R. L. Gregory (ed.), *The Oxford companion to the mind* (2nd ed., pp. 698–703). Oxford, UK: Oxford University Press.
- Jain, A., Duin, R., & Mao, J. (2000). Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22, 4–37.
- James, T. W., Kim, S., & Fisher, J. S. (2007). The neural basis of haptic object processing. *Canadian Journal of Experimental Psychology*, 61(3), 219–229.
- Jarc, D. J., Feldman, M. B., & Heller, R. S. (2000). Assessing the benefits of interactive prediction using web-based algorithm animation courseware. *ACM SIGCSE Bulletin*, 32(1), 377–381.
- Jonassen, D. H., Peck, K. L., & Wilson, B. G. (1999). *Learning with technology: A constructivist perspective*. Upper Saddle River, NJ: Merrill.
- Kagan, J. (2009). *The three cultures: Natural sciences, social sciences, and the humanities in the 21st century*. Cambridge University Press.
- Kalyuga, S., Chandler, P., & Sweller, J. (1999). Managing spilt attention and redundancy in multimedia instruction. *Applied Cognitive Psychology*, 13, 351–371.
- Kaminski, K., Switzer, J., & Gloeckner, G. (2009). Workforce readiness: A study of university students' fluency with information technology. *Computers & Education*, 53(2), 228–233.
- Karavirta, V., & Shaffer, C. A. (2015). Creating engaging online learning material with the JSAV JavaScript algorithm visualization library. *IEEE Transactions on Learning Technologies*, 9(2), 171–183.
- Kátai, Z. (2011). Multi-sensory method for teaching-learning recursion. *Computer Applications in Engineering Education*, 19(2), 234–243.
- Kátai, Z. (2014a, June). Selective hiding for improved algorithmic visualization. In: *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education* (pp. 33–38). ACM.
- Kátai, Z. (2014b, June). Intercultural computer science education. In: *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education* (pp. 183–18). ACM.
- Kátai, Z. (2014c, June). Algorithmic thinking for ALL: A motivational perspective. In: *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education* (pp. 353–353). ACM.
- Kátai, Z. (2015). The challenge of promoting algorithmic thinking of both sciences- and humanities-oriented learners. *Journal of Computer Assisted Learning*, 31(4), 287–299.
- Kátai, Z. (2020). Promoting computational thinking of both sciences- and humanities-oriented students: An instructional and motivational design perspective. *Educational Technology Research and Development*, 68(5), 2239–2261.

- Káta, Z., Juhász, K., Adorjáni, A. K. (2008). On the role of senses in education. *Computers & Education*, 51(4), 1707–1717.
- Káta, Z., Osztián, E., Osztián, P. R., Nagy, J. E., & Cosma, C. (2020). *AlgoRythmics* (Version 2.2.0). Sapientia Hungarian University of Transylvania. <https://algorhythmics.ms.sapientia.ro/>.
- Káta, Z., Osztián, E., Osztián, P. R., & Vekov, G. K. (2018, January 30). *AlgoRythmics* [Video]. Youtube. <https://www.youtube.com/user/AlgoRythmics/>.
- Káta, Z., & Tóth, L. (2010). Technologically and artistically enhanced multi-sensory computer-programming education. *Teaching and Teacher Education*, 26, 244–251.
- Káta, Z., & Tóth, L. (2011, March 29). *AlgoRythmics* [Video]. YouTube. <https://www.youtube.com/user/AlgoRythmics>.
- Kavanaugh, J. F. (Ed.). (1991). *The language continuum from infancy to literacy*. Baltimore: York.
- Keller, J., & Suzuki, K. (2004). Learner motivation and e-learning design: A multinationally validated process. *Journal of Educational Media*, 29(3), 229–239.
- Keller, J. M. (1983). Motivational design of instruction. *Instructional design theories and models: An overview of their current status*, 1(1983), 383–434.
- Keller, J. M. (1987). Development and use of the ARCS model of instructional design. *Journal of instructional development*, 10(3), 2.
- Khan, A., Ahmad, F. H., & Malik, M. M. (2017). Use of digital game based learning and gamification in secondary school science: The effect on student engagement, learning and gender difference. *Education and Information Technologies*, 22(6), 2767–2804.
- Knuth, D. E. (1985). Algorithmic thinking and mathematical thinking. *The American Mathematical Monthly*, 92(3), 170–181.
- Korhonen, A., & Malmi, L. (2000, July). Algorithm simulation with automatic assessment. In: *Proceedings of the 5th Annual SIGCSE/SIGCUE ITiCSE Conference on Innovation and Technology in Computer Science Education* (pp. 160–163). ACM Press, New York.
- Krathwohl, D. R. (2002). A revision of Bloom’s taxonomy: An overview. *Theory into practice*, 41(4), 212–218.
- Lawrence, A. W. (1993). *Empirical studies of the value of algorithm animation in algorithm understanding*. Atlanta. <http://www.dtic.mil>.
- Lee, O., & Anderson, C. W. (1993). Task engagement and conceptual change in middle school science classrooms. *American educational research journal*, 30(3), 585–610.
- Lee, O., & Brophy, J. (1996). Motivational patterns observed in sixth-grade science classrooms. *Journal of Research in Science Teaching*, 33(3), 303–318.
- Lepper, M. R., Corpus, J. H., & Iyengar, S. S. (2005). Intrinsic and extrinsic motivational orientations in the classroom: Age differences and academic correlates. *Journal of educational psychology*, 97(2), 184.

- Lepper, M. R., & Malone, T. W. (1987). Intrinsic motivation and instructional effectiveness in computer-based education. *Aptitude, learning, and instruction*, 3, 255–286.
- Lewandowski, G., Bouvier, D. J., Chen, T. Y., McCartney, R., Sanders, K., Simon, B., & VanDeGrift, T. (2010). Commonsense understanding of concurrency: Computing students and concert tickets. *Communications of the ACM*, 53(7), 60–70.
- Lister, R., Adams, E., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M. et al. (2004). A multinational study of reading and tracing skills in novice programmers. In: *Working group reports from ITiCSE'04 on innovation and technology in computer science education*. New York, NY, USA: ACM Press.
- Litman, J. (2005). Curiosity and the pleasures of learning: Wanting and liking new information. *Cognition & Emotion*, 19(6), 793–814.
- Loftin, R. B., Brooks, F. P., & Dede, C. (1998). Virtual reality in education: Promise and reality. In: *Proceedings of the virtual reality annual international symposium*.
- Lu, J. J., & Fletcher, G. H. (2009). Thinking about computational thinking. *ACM SIGCSE Bulletin*, 41(1), 260–264.
- Lumpkin, A., Achen, R. M., & Dodd, R. K. (2015). Student perceptions of active learning. *College Student Journal*, 49(1), 121–133.
- Malone, T. W., & Lepper, M. R. (1987). Making learning fun: A taxonomy of intrinsic motivations for learning. *Aptitude, learning, and instruction*, 3, 223–253.
- Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014, June). Computational thinking in K–9 education. In: *Proceedings of the working group reports of the 2014 on innovation & technology in computer science education conference* (pp. 1–29). ACM.
- Martens, R., Gulikers, J., & Bastiaens, T. (2004). The impact of intrinsic motivation on e-learning in authentic computer tasks. *Journal of computer assisted learning*, 20(5), 368–376.
- May, S. (1994). *Making multicultural education work*. Clevedon: Multilingual Matters.
- Mayer, R. E. (1999). Designing instruction for constructivist learning. *Instructional-design theories and models: A new paradigm of instructional theory*, 2, 141–159.
- Mayer, R. E. (2003). *Learning and Instruction*. Upper Saddle River, NJ: Prentice Hall.
- Mayer, R. E., & Chandler, P. (2001). When learning is just a click away: Does simple interaction foster deeper understanding of multimedia messages? *Journal of Educational Psychology*, 93(2), 390–397.
- Mayer, R. E., & Moreno, R. (2003). Nine ways to reduce cognitive load in multimedia learning. *Educational psychologist*, 38(1), 43–52.

- McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y. B. D. et al. (2001). A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. In: *Working group reports from ITiCSE'01 on Innovation and Technology in Computer Science Education*, Canterbury, UK.
- McIlroy, D., Bunting, B., Tierney, K., & Gordon, M. (2001). The relation of gender and background experience to self-reported computing anxieties and cognitions. *Computers in Human Behavior*, 17(1), 21–33.
- Mead, J., Gray, S., Hamer, J., James, R., Sorva, J., Clair, C. S. et al. (2006). A cognitive approach to identifying measurable milestones for programming skill acquisition. In: *Working group reports from ITiCSE'06 on Innovation and Technology in Computer Science Education*. Bologna, Italy: ACM Press.
- Meij, J., Jong, T. (2006). Supporting students' learning with multiple representations in a dynamic simulation-based learning environment. *Learning and Instruction*, 16(3), 199–212.
- Meredith, A. M. (2002). On the neuronal basis for multisensory convergence: A brief overview. *Cognitive Brain Research*, 14, 31–40.
- Messick, S. (1976). Personality consistencies in cognition and creativity. In: S. Messick (ed.), *Individuality in learning* (pp. 4–23). San Francisco: Jossey-Bass.
- Milkova, E. (2005). Developing of algorithmic thinking: The base of programming. *International Journal of Continuing Engineering Education and Life-Long Learning*, 15(3–6), 135–147.
- Moelwyn-Hughes, L. (2003, Autumn). Dancing the words. *Animated: The community dance magazine*. <http://www.communitydance.org.uk>.
- Moen, J. (2008). Can movement be virtual? *Keho*, (3), 23–24.
- Moreno, R., Ozogul, G., & Reisslein, M. (2011). Teaching with concrete and abstract visual representations: Effects on students' problem solving, problem representations, and learning perceptions. *Journal of Educational Psychology*, 103, 32–47.
- Mork, S. M. (2011). An interactive learning environment designed to increase the possibilities for learning and communicating about radioactivity. *Interactive Learning Environments*, 19(2), 163–177.
- Mtangi, S. (2006). Liz Lerman Exchange connects science and dance. *The Wesleyan Argus*, CXLII(26). <http://wesleyanargus.com/2006/02/10/liz-lerman-exchange-connects-science-and-dance>.
- Myller, N., Laakso, M., & Korhonen, A. (2007, June). Analyzing engagement taxonomy in collaborative algorithm visualization. In: *Proceedings of the 12th annual SIGCSE Conference on Innovation and Technology in Computer Science Education* (pp. 251–255).

- Nagy, E. J., Osztián, P. R., Cosma, C., Káta, Z., & Osztián, E. (2019, June). Looking for the Optimal Interactivity Level in the AlgoRythmics Learning Environment. In: *EdMedia+ Innovate Learning* (pp. 106–114). Association for the Advancement of Computing in Education (AACE).
- Nakamura, J., & Csíkszentmihályi, M. (2002). The concept of flow. *Handbook of positive psychology* (Snyder, C. R., & Shane, J. L. (eds.)), 89–105.
- Naps, T. L., Eagan, J. R., & Norton, L. L. (2000, March). JHAVÉ—An environment to actively engage students in Web-based algorithm visualizations. In: *Proceedings of the Thirty-First SIGCSE Technical Symposium on Computer Science Education* (pp. 109–113).
- Naps, T. L., Röbling, G., Almstrum, V., Dann, W., Fleischer, R., Hundhausen, C., & Velázquez-Iturbide, J. Á. (2002). Exploring the role of visualization and engagement in computer science education. In: *Working group reports from ITiCSE on Innovation and Technology in Computer Science Education* (pp. 131–152).
- National Council for Curriculum and Assessment (NCCA). (2005). *Intercultural Education in Primary Schools*. www.ncca.ie/uploadedfiles/publications/Intercultural.pdf.
- Navrat, P. (1994). Hierarchies of programming concepts: Abstraction, generality, and beyond. *SIGCSE Bulletin*, 26(3).
- Nelson-Barber, S. & Estrin, E. (1995). Bringing native American perspectives to mathematics and science teaching. *Theory into Practice*, 34(3), 174–185.
- Nemirow, L. E. (1995). Understanding rules. *The Journal of Philosophy*, 92(1), 28–43.
- Newell, A. (1980). Physical symbol systems. *Cognitive Science*, 4(2), 135–183.
- Nieto, S. (1992). *Affirming diversity: The sociopolitical context of Multicultural Education*. New York: Longman.
- Nugteren, M. L., Tabbers, H. K., Scheiter, K., & Paas, F. (2014). Simultaneous and sequential presentation of realistic and schematic instructional dynamic visualizations. In: *Handbook of human centric visualization* (pp. 605–622). Springer, New York, NY.
- O’Toole, J. M., & Absalom, D. (2003). The impact of blended learning on student outcomes: Is there room on the horse for two? *Journal of Educational Media*, 28(2–3), 179–190.
- Omrod, J. E. (2002). *Educational Psychology* (4th ed.). Upper Saddle River, NJ: Prentice Hall.
- Oracle Education Foundation (OEF) (2001). *Out of Memory*. ThinkQuest Internet Challenge. <http://library.thinkquest.org/C0110291/science/factors/senses.php>.
- Oreck, B. (2004). The artistic and professional development of teachers. A study of teachers’ attitudes toward and use of the arts in teaching. *Journal of Teacher Education*, 55(1), 55–69.

- Osztaián, P. R., Kátai, Z., & Osztaián, E. (2020). Algorithm visualization environments: Degree of interactivity as an influence on student-learning. In: *50th Annual Frontiers in Education Conference*. In press.
- Ott, M., & Tavella, M. (2010). Motivation and engagement in computer-based learning tasks: Investigating key contributing factors. *World Journal on Educational Technology*, *2*(1), 1–15.
- Overy, K., & Turne, B. (2009). The rhythmic brain. *Cortex*, *45*(1), 1–3. doi:10.1016/j.cortex.2008.11.002.
- Paas, F., Renkl, A., & Sweller, J. (2003). Cognitive load theory and instructional design: Recent developments. *Educational psychologist*, *38*(1), 1–4.
- Paas, F., & Sweller, J. (2012). An evolutionary upgrade of cognitive load theory: Using the human motor system and collaboration to support the learning of complex cognitive tasks. *Educational Psychology Review*, *24*(1), 27–45.
- Papert, S. (1996). An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning*, *1*(1), 95–123.
- Papert, S. A. (1981). *Mindstorms: Children, computers, and powerful ideas*. Basic Books.
- Parrish, P. E. (2009). Aesthetic principles for instructional design. *Educational Technology Research and Development*, *57*(4), 511–528.
- Pedró, F. (2006). *The New Millennium learners: Challenging our views on ICT and learning*. OECD-CERI. <http://www.oecd.org>.
- Philpot, T. A., Hall, R. H., Hubing, N., & Flori, R. E. (2005). Using games to teach statics calculation procedures: Application and assessment. *Computer Applications in Engineering Education*, *13*(3), 222–232.
- Plass, J. L., Homer, B. D., & Hayward, E. O. (2009). Design factors for educationally effective animations and simulations. *Journal of Computing in Higher Education*, *21*(1), 31–61.
- Pobric, G., Mashal, N., Faust, M., & Lavidor, M. (2008). The role of the right cerebral hemisphere in processing novel metaphoric expressions: A transcranial magnetic stimulation study. *Journal of Cognitive Neuroscience*. *20*(1), 170–181.
- Polya, G. (1945). *How to solve it?* Princeton, NJ: Princeton University Press.
- Procter, C. (2003). Blended learning in practice. In: *Proceedings of Conference on Education in a Changing Environment*. UK: University of Salford. http://www.ece.salford.ac.uk/proceedings/papers/cp_03.rtf.
- Rau, M. A., Michaelis, J. E., & Fay, N. (2015). Connection making between multiple graphical representations: A multi-methods approach for domain-specific grounding of an intelligent tutoring system for chemistry. *Computers & Education*, *82*, 460–485.
- Rawson, M. B. (1992). *The many faces of dyslexia*. Baltimore: International Dyslexia Association.

- Rheinberg, F., Vollmeyer, R., & Burns, D. A. (2001). QCM: A questionnaire to assess current motivation in learning situations. *Diagnostica*, 47, 57–66.
- Riaz, S., Rambli, D. R. A., Salleh, R., & Mushtaq, A. (2011). Exploratory factor analysis (EFA) to examine learner's aesthetic perceptions and motivation through their aesthetic-emotions in informal visual environments. *Information Technology Journal*, 10(7), 1268–1284.
- Riffell, S., & Sibley, D. (2005). Using web-based instruction to improve large undergraduate biology courses: An evaluation of hybrid course format. *Computers & Education*, 44(3), 217–235.
- Robertson, J., & Howells, C. (2008). Computer game design: Opportunities for successful learning. *Computers & Education*, 50(2), 559–578.
- Román-González, M., Pérez-González, J. C., Moreno-León, J., & Robles, G. (2018). Can computational talent be detected? Predictive validity of the computational thinking test. *International Journal of Child-Computer Interaction*, 18, 47–58.
- Rovai, A., Ponton, M., Wighting, M., & Baker, J. (2007). A comparative analysis of student motivation in traditional classroom and elearning courses. *International Journal on E-Learning*, 6(3), 413–432.
- San Diego State University—Educational Technology (SDSU-ET) (2008). Sense and memory. *Encyclopedia of Educational Technology (EET)*. <http://coe.sdsu.edu/eet>.
- Schaffer, K., Stern, E., & Kim, S. (2001). *Math dance with Dr. Schaffer and Mr. Stern* (Prelim. ed.). Santa Cruz, CA: MovespeakSpin. <http://www.mathdance.org>.
- Scheiter, K., Gerjets, P., Huk, T., Imhof, B., & Kammerer, Y. (2009). The effects of realism in learning with dynamic visualizations. *Learning and Instruction*, 19(6), 481–494.
- Schnotz, W., & Rasch, T. (2005). Enabling, facilitating, and inhibiting effects of animations in multimedia learning: Why reduction of cognitive load can have negative results on learning. *Educational Technology Research and Development*, 53(3), 47.
- Schrader, C., Brich, J., Frommel, J., Riemer, V., & Rogers, K. (2017). Rising to the challenge: An emotion-driven approach toward adaptive serious games. In: *Serious games and edutainment applications* (pp. 3–28). Springer, Cham.
- Schroeder, C. E., & Foxe, J. J. (2002). The timing and laminar profile of converging inputs to multisensory areas of the macaque neocortex. *Cognitive Brain Research*, 14, 187–198.
- Schwan, S., & Riempp, R. (2004). The cognitive benefits of interactive videos: Learning to tie nautical knots. *Learning and instruction*, 14(3), 293–305.
- Schwank, I. (1993). On the analysis of cognitive structures in algorithmic thinking. *Journal of Mathematical Behavior*, 12(2), 209–231.

- Selby, C., & Woollard, J. (2013). *Computational thinking: The developing definition*. University of Southampton. <https://eprints.soton.ac.uk/id/eprint/356481>.
- Shaffer, C. A., Cooper, M. L., Alon, A. J. D., Akbar, M., Stewart, M., Ponce, S., & Edwards, S. H. (2010). Algorithm visualization: The state of the field. *ACM Transactions on Computing Education (TOCE)*, 10(3), 1–22.
- Shaffer, C. A., Cooper, M., & Edwards, S. H. (2007, March). Algorithm visualization: A report on the state of the field. In: *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education* (pp. 150–154). ACM Press, New York.
- Shams, L., & Seitz, A. R. (2008). Benefits of multisensory learning. *Trends in Cognitive Sciences*, 12(11), 411–417.
- Shaywitz, S. (2003). *Overcoming dyslexia: A new and complete science-based program for overcoming reading problems at any level*. New York: Knopf.
- Shen, Q., Chung, J. K. H., Challis, D., & Cheung, R. C. T. (2007). A comparative study of student performance in traditional mode and online mode of learning. *Computer Applications in Engineering Education*, 15(1), 30–40.
- Shimojo, S., & Shams, L. (2001). Sensory modalities are not separate modalities: Plasticity and interactions. *Current Opinion in Neurobiology*, 11, 505–509.
- Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22, 142–158.
- Simon, B., Chen, T. Y., Lewandowski, G., McCartney, R., & Sanders, K. (2006, September). Commonsense computing: What students know before we teach (episode 1: sorting). In: *Proceedings of the Second International Workshop on Computing Education Research* (pp. 29–40). ACM.
- Simon, B., Kinnunen, P., Porter, L., & Zazkis, D. (2010, June). Experience report: CS1 for majors with media computation. In: *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education* (pp. 214–218). ACM.
- Singh, H., & Reed, C. (2001). *A white paper: Achieving success with blended learning*. Los Angeles: Centra Software.
- Sizemore, B. A. (1981). The politics of multicultural education. *Urban Education*, 5, 4–11.
- Sleeter, C. E. (1996). *Multicultural education as social activism*. Albany: State University of New York Press.
- Snow, C. P. (1959). *The two cultures and the scientific revolution*. Cambridge University Press.
- Soloway, E., Bonar, J., & Ehrlich, K. (1983). Cognitive strategies and looping constructs: An empirical study. *Communications of the ACM*, 26(11).
- Spangenberg, T. H., & Brynskov, M. (2018). The nature of computational thinking in computing education. *International Journal of Information and Education Technology*, 8(10), 742–747.

- Spohrer, J., & Soloway, E. (1986). Novice mistakes: Are the folks wisdoms correct? *Communications of the ACM*, 29(7).
- Staley, J. D. (2006, June 6). *Imagining the multisensory classroom*. Campus Technology. <http://campustechnology.com/articles/40941>.
- Stein, B. E., & Meredith, M. A. (1993). *The merging of the senses*. London: MIT Press.
- Stenning, K. (1998). Distinguishing semantic from processing explanations of usability of representations: Applying expressiveness analysis to animation. In: Lee, J. (ed.), *Intelligence and multimodality in multimedia interfaces: Research and applications*. Cambridge; MA: AAAI Press.
- Stevens, J., & Goldberg, D. (2001). *For the learners' sake: A practical guide to transform your classroom and school*. Tucson, AZ: Zephyr Press.
- Sundar, S. S., Kalyanaraman, S., & Brown, J. (2003). Explicating website interactivity: Impression-formation effects in political campaign sites. *Communication Research*, 30(1), 30–59.
- Swenson, P., & Evans, M. (2003). Hybrid courses as learning communities. In: Reisman, S. (ed.), *Electronic learning communities issues and practices* (pp. 27–72). Greenwich, CT: Information Age Publishing.
- Tate, W. (1995). Returning to the root: A culturally relevant approach to mathematics pedagogy. *Theory into Practice*, 34(3), 166–173.
- Tedre, M., & Denning, P. J. (2016, November). The long quest for computational thinking. In: *Proceedings of the 16th Koli Calling International Conference on Computing Education Research* (pp. 120–129).
- Tew, A. E., McCracken, W. M., & Guzdial, M. (2005, October). Impact of alternative introductory courses on programming concept understanding. In: *Proceedings of the First International Workshop on Computing Education Research* (pp. 25–35). ACM.
- The dance of mathematics. (2006). New Zealand Institute of Mathematics & its Applications. http://www.mathsreach.org/pdfs/06Applied_algorithms.pdf.
- Thomas, E., & Mulvey, A. (2008). Using the arts in teaching and learning. Building student capacity for community-based work in health psychology. *Journal of Health Psychology*, 13(2), 239–250.
- Thompson, S. M. (2003). *Multisensory learning in inclusive classrooms*. Academic Exchange Quarterly.
- Thorson, K. S., & Rodgers, S. (2006). Relationships between blogs as eWOM and interactivity, perceived interactivity, and parasocial interaction. *Journal of Interactive Advertising*, 6(2), 5–44.
- Tofade, T., Elsner, J., & Haines, S. T. (2013). Best practice strategies for effective use of questions as a teaching tool. *American Journal of Pharmaceutical Education*, 77(7), 155.

- Triberti, S., Chirico, A., La Rocca, G., & Riva, G. (2017). Developing emotional design: Emotions as cognitive processes and their role in the design of interactive technologies. *Frontiers in Psychology*, 8, 1773.
- TPUB: Integrated publishing. (2020). *Navy Instructional Theory* (pp. 25–26). <http://navyadministration.tpub.com/>.
- Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 59(236), 433–460.
- Turner, J. C., & Meyer, D. K. (2004). A classroom perspective on the principle of moderate challenge in mathematics. *The Journal of Educational Research*, 97(6), 311–318.
- Tversky, B., Morrison, J. B., & Betrancourt, M. (2002). Animation: can it facilitate? *International Journal of Human-Computer Studies*, 57(4), 247–262.
- Um, E. R., Song, H., & Plass, J. (2007, June). The effect of positive emotions on multimedia learning. In: *EdMedia+ Innovate Learning* (pp. 4176–4185). Association for the Advancement of Computing in Education (AACE).
- Urquiza-Fuentes, J., & Velázquez-Iturbide, J. Á. (2009). A survey of successful evaluations of program visualization and algorithm animation systems. *ACM Transactions on Computing Education (TOCE)*, 9(2), 1–21.
- US-BLS (US Bureau of Labor Statistics). (2020). <http://www.bls.gov/ooh/>.
- Valdez, A. (1999). *Learning in living colour: Using literature to incorporate multicultural education into the primary curriculum*. Boston: Allyn and Bacon.
- Valiathan, P. (2002). Blended learning models. *Learning Circuits*, 3(8), 50–59.
- Valle, A., Cabanach, R. G., Rodríguez, S., Nuñez, J. C., González-Pienda, J. A., Solano, P., & Rosário, P. (2011). A motivational perspective on the self-regulated learning in higher education. *Journal of Education Research*, 5.
- Van Gendt, K. & Verhagen, P. (2001, November). *Visual testing. Searching for guidelines*. Paper presented at the 24th National Convention of the Association for Educational Communications and Technology, Atlanta, Georgia.
- Van Gog, T., Paas, F., Marcus, N., Ayres, P., & Sweller, J. (2009). The mirror neuron system and observational learning: Implications for the effectiveness of dynamic visualizations. *Educational Psychology Review*, 21(1), 21–30.
- van Wassenhove, V., Grant, K.W., & Poeppel, D. (2005). Visual speech speeds up the neural processing of auditory speech. *Proceedings of the National Academy of Sciences*, 102(4), 1181–1186.
- Vollmeyer, R., Burns, B. D., & Rheinberg, F. (2000). Goal specificity and learning with a multimedia program. In: *Proceedings of the Annual Meeting of the Cognitive Science Society* 22(22).
- Voto, D., Viñas, L. M., & D’Auria, L. (2005). Multisensory interactive installation. In: *Sound and Music Computing ‘05, XV CIM*. November 24–26, Salerno, Italy.

- Wang, Q. (2009). Designing a web-based constructivist learning environment. *Interactive Learning Environments*, 17(1), 1–13.
- Ward, H., Hewlett, C., Roden, J., & Foreman, J. (2005). *Teaching science in the primary classroom: A practical guide*. London: SAGE Publications Ltd.
- Watters, J. J. (2010). Career decision making among gifted students: The mediation of teachers. *Gifted Child Quarterly*, 54(3), 222–238.
- Wechsler, R. (1997). Computers and art: A dancer's perspective. *IEEE Technology and Society Magazine*, 16(3), 7–14.
- Weiner, B. (1990). History of motivational research in education. *Journal of Educational Psychology*, 82(4), 616.
- Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems*. Los Altos, CA: Morgan Kaufmann.
- West, T. G. (1994). Advanced interaction: A return to mental models and learning by doing. *Computers & Graphics*, 18(5), 685–689.
- Wilhelm, R. W. (1994). Exploring the practice-rhetoric gap: Current curriculum for African-American history month in some Texas elementary schools. *Journal of Curriculum and Supervision*, 9(2), 217–223.
- Wilson, B. (1981). The cultural contexts of science and mathematics education: Preparation of a bibliographic guide. *Studies in Science Education*, 8(1), 27–44.
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.
- Wing, J. M. (2010). *Computational thinking: What and why?* Unpublished manuscript. Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, <https://www.cs.cmu.edu/CompThink/resources/TheLinkWing.pdf>.
- Winslow, E. L. (1996). Programming pedagogy – A psychological overview. *SIGCSE Bulletin*, 28(3), 17–22.
- Witkin, H. A., Moore, C. A., Goodenough, D. R., & Cox, P. W. (1977). Field dependent and field independent cognitive styles and their educational implications. *Review of Educational Research*, 47, 1–64.
- Wittgenstein, L. (2009). *Philosophical investigations*. John Wiley & Sons.
- Wlodkowski, R. J. (1985). *Enhancing adult motivation to learn*. San Francisco: Jossey-Bass.
- Wong, T., Bigras, P., & Cervera, D. (2005). A software application for visualizing and understanding hydraulic and pneumatic networks. *Computer Applications in Engineering Education*, 13(3), 169–180.
- Wood, Z. J., Muhl, P., & Hicks, K. (2016, February). Computational art: Introducing high school students to computing via art. In: *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (pp. 261–266). ACM.

- Wu, M. L., & Richards, K. (2011). Facilitating computational thinking through game design. In: M. Chang, W. Y. Hwang, M. P. Chen, & W. Müller (eds.), *Proceedings of Edutainment Technologies. Educational Games and Virtual Reality/Augmented Reality Applications* (pp. 220–227). Heidelberg: Springer Berlin.

APPENDIX

The questionnaires used in Study 8 (*Chapter 10*)

Questionnaire 1

1. I understand the concept of algorithm.
2. I understand what a sorting algorithm means.
3. I am totally unfamiliar with this subject.
4. I agree that algorithmic thinking is a valuable ability that I should possess.
5. I agree that understanding how sorting algorithms work can enhance my algorithmic thinking.

Questionnaire 2

1. (E) I like the idea behind this e-learning session.
2. (E) I consider that combining arts (folk dances) with sciences (computer algorithms) is an interesting idea.
3. (E) I consider that combining something so modern (computer sciences) with something so traditional (folk dances) is a surprising idea.
4. (E) I am curious about how sorting algorithms work.
5. (E) I am curious about this e-learning session because of the dances.
6. (E) I am curious about this e-learning session because of the art–science combination.
7. I understood what aspects of the choreography I should focus on to identify:
 - the comparing and swapping operations,
 - the traverses of the number sequence,
 - the sorting strategy.
8. I understood that this e-learning session requires a substantial active participation from me: to understand, reconstruct, and orchestrate the studied sorting algorithm.
9. (C) Identification of a danced strategy must be a non-trivial task.
10. (C) I am afraid that what this e-learning session implies is beyond me.
11. (I) The idea of reconstructing and orchestrating an algorithm attracts me.

Questionnaire 3

1. (E) I liked this phase of the e-learning session.
2. (E) I enjoyed the dance performance.
3. (E) The dance choreography helped me to imagine what a sorting algorithm means.
4. (E) I did not realize what was going on.
5. (E) I grasped the logic of the sorting strategy.
6. (C) This e-learning session seems to be a challenging one.
7. (C) I hope the computer animation will help me to understand the algorithm more deeply.
8. (I) The idea that I will have to try to orchestrate such a complex strategy scares me.
9. (I) All my confidence is in the help button.

Questionnaire 4

1. (E) I liked this phase of the e-learning session.
2. (E) The animation was expressive and interesting.
3. (E) I did not realize what was going on.
4. (E) I realized the correspondences between the dance choreography and the computer animation (comparing, swapping, traverses ...).
5. (E) I grasped the logic of the sorting strategy.
6. I understood what to reconstruct/orchestrate (using the mouse) an algorithm means:
 - to choose (according to the strategy of the algorithm):
 - the next pair of elements to be processed,
 - the proper operation (comparing or swapping) to be applied.
7. I understood what the role of the help button is:
 - to inform me about the next step to be performed (what operation on what elements has to be applied).
8. I understood that my performance is measured by:
 - how many faults (wrong pair of elements or wrong operation) I have made,
 - how many times I have used the help button.
9. (I) I appreciate that the software makes possible to “do” what I have just “heard” and “seen”.
10. (C) Reconstructing the same operation sequence does not seem to be a complicated task.
11. (I) I appreciate that a help button has been implemented to take me through the possible stuck points.
12. (C) I hope I will not need to use the help button in the reconstruction process.

Questionnaire 5

1. (E) I liked this phase of the e-learning session.
2. (E) I like the way the software implements the principle of “hearing, seeing, and doing”.
3. (E) I am positively surprised that I have made only a few faults.
4. (E) I often applied guessing.
5. (E) My memory helped me in the reconstruction process.
6. (E) I have achieved this performance because I understood the strategy the algorithm applies.
7. (E) I am disappointed that I was forced to use the help button so many times.
8. (E) It would have been better if I had made use of the help button more efficiently.
9. (E) By “doing the algorithm”, I understand it better.
10. (I) I appreciate that the software makes it possible to choose between repeating the current-level task and moving to the next-level task.
11. (C) I would like to repeat this task until I manage to perform it faultlessly.
12. (C) Although I made some mistakes, it would be boring to try it again.
13. (I) I appreciate that the software makes it possible to orchestrate the algorithm on randomly generated input sequences.
14. (C) Orchestrating this strategy on random input sequences will be a challenging task.
15. (C) I consider that what this e-learning session implies is beyond me.
16. (I) I appreciate that a help button has been implemented to take me through the possible stuck points.
17. (C) I think I will not need to use the help button in orchestrating the algorithm on any input sequence.

Questionnaire 6

1. (E) I liked this phase of the e-learning session.
2. (E) I like the way the software implements the principle of progressive difficulty.
3. (E) I am positively surprised that I made only a few faults.
4. (E) I often applied guessing.
5. (E) I have achieved this performance because I understood the strategy the algorithm applies.
6. (E) I am disappointed that I was forced to use the help button so many times.
7. (E) It would have been better if I had made use of the help button more efficiently.

8. (E) Orchestrating the algorithm I identified lacunas in my understanding of the algorithm.
9. I understood that although the numbers will be hidden, I will be informed about the results of the comparing operations.
10. (I) I appreciate that the software makes it possible to choose between repeating the current-level task and moving to the next-level task.
11. (C) I would like to repeat this task until I manage to perform it faultlessly.
12. (C) Although I made some mistakes, it would be boring to try it again.
13. (I) I appreciate that the software makes it possible to test the solidity of my understanding regarding the algorithm on hidden sequences.
14. (C) Since I grasped the logic of the strategy, it is no problem for me to orchestrate it on hidden sequences.
15. (C) I consider that what this e-learning session implies is beyond me.
16. (I) I appreciate that a help button has been implemented to bring me through the possible stuck points.
17. (C) I hope that I will be able to perform the “black-box orchestration process” without using the help button.

Questionnaire 7

1. (E) I liked this phase of the e-learning session.
2. (E) I like the way the software implements the principle of progressive difficulty.
3. (E) I am positively surprised that I made only a few faults.
4. (E) I often applied guessing.
5. (E) I have achieved this performance because I understood the strategy the algorithm applies.
6. (E) I am disappointed that I was forced to use the help button so many times.
7. (E) It would have been better if I had made use of the help button more efficiently.
8. (E) Orchestrating the algorithm on a hidden sequence revealed further lacunas in my understanding of the algorithm.
9. (I) I appreciate that the software makes it possible to choose between repeating the current-level task and moving to the next-level task.
10. (C) I would like to repeat this task until I manage to perform it faultlessly.
11. (C) Although I made some mistakes, it would be boring to try it again.
12. (C) I consider that what this e-learning session implies is beyond me.
13. (E) I am curious to see the parallel illustration of several sorting algorithms on colour scale bars.

Questionnaire 8

1. (E) I liked this last phase of the e-learning session.
2. (E) The parallel simulation was expressive.
3. (E) The colourful global image the parallel simulation generated was nice.
4. (E) I am curious about the other sorting algorithms, too.

Questionnaire 9

1. (E) I liked this e-learning session.
2. (E) I liked this e-learning session because of its interestingness.
3. (E) I liked this e-learning session because of its challenge.
4. (E) I liked this e-learning session because of its interactivity.
5. (E) I liked this e-learning session because of the dances.
6. (E) I consider this e-learning experience to be a useful one because it initiated me into what sorting algorithms mean.
7. (E) I consider this e-learning experience to be a useful one because it resulted in a deeper understanding of how the studied algorithm works.
8. (E) I consider this e-learning experience to be a useful one because it may contribute to a developed algorithmic thinking.
9. (E) I would like to study the other sorting algorithms the ending parallel simulation presented.
10. (E) I would recommend this e-learning experience for my friends, too.

KIVONAT

ALGORYTHMICS: MŰVÉSZI ELEMekkel TARKÍTOTT E-LEARNING KÖRNYEZET INFORMATIKAOKTATÁSHOZ

A könyv az AlgoRythmics projekt több mint 15 évre visszatekintő történetét göngyölíti fel. Bemutatja a kutatási csoportot, valamint nyolc kiemelkedő kutatást, amely ebben a tanulási környezetben zajlott. Mindenik kutatás kapcsán részletezzük a szakirodalmi háttérrel, az implementált kísérletet vagy kísérleteket, illetve az elért eredményeket. Két fejezet is foglalkozik magával a környezettel, annak eredeti és megújult változatával. A befejező részek a futó projektjeinkből adnak ízelítőt, és jövőbeli terveinket körvonalazzák.

REZUMAT

ALGORYTHMICS: ALGORITMICĂ ASISTATĂ DE CALCULATOR ÎMBINATĂ CU ELEMENTE ARTISTICE

Cartea descrie istoria de mai bine de 15 ani a proiectului AlgoRythmics, prezintă grupul de cercetare din spatele proiectului și detaliază opt studii care au fost implementate în acest mediu de învățare. Vizavi de fiecare cercetare, prezintă contextul științific aferent, experimentul efectuat și rezultatele obținute. Două capitole se concentrează pe aplicația AlgoRythmics, varianta inițială și cea revizuită. Ultimele capitole conțin o descriere succintă a proiectelor actuale ale grupului de cercetare AlgoRythmics, și prefigurează liniile de cercetare planificate pentru viitor.

ABOUT THE AUTHOR

Zoltán Kátai started his professional career (1992–) as an informatics teacher in secondary education. Presently (2001–), he is Associate Professor (2014–) at Sapientia Hungarian University of Transylvania (Department of Mathematics and Informatics), Târgu-Mureş, Romania. His research areas include computer science education, computer-assisted instruction, dynamic programming, and graph algorithms. He is the initiator and leader of the AlgoRythmics project, which in 2013 won the *Best Practices in Education* award offered by Informatics Europe.

Scientia Publishing House
400112 Cluj-Napoca
Matei Corvin street 4.
Tel./fax: +40-364-401454
E-mail: scientia@kpi.sapientia.ro
www.scientiakiado.ro

English Proofreading:
István Szász-Köpeczy

Layout Editor:
Metaforma Ltd.

Cover: Tipotéka Ltd.
Cover art: Attila Nagy

Typography:
Elemér Könczey

Printed by F&F International Ltd.
Director: Enikő Ambrus

A major responsibility of educational systems in the 21st century is to prepare future generations for the challenges involved with the increasing computerization of our everyday lives and to meet the demands of one of the fastest-growing job markets: computing. The goal of our beloved AlgoRythmics project is to promote computing education for all by taking into account the key elements from the most relevant computational thinking definitions. For this purpose, we have created an engaging algorithm visualization environment that is built around a collection of interactive dynamic visualizations illustrating basic computer algorithms.

Making computing education attractive for different categories of learners is a challenging initiative. A possible approach might be contextualization. The AlgoRythmics learning environment has been designed along this approach. Since music and dance are relatively close to most people, this environment visualizes searching and sorting algorithms by professional dance choreographies (folkdance, flamenco, ballet). The “dance floor” we have created is an interactive and intuitive user interface which guides learners from dance to code. From the perspective of the teaching-learning process, the most important features of the environment are its unified, artistically enhanced, human-movement-effect-enriched, multisensory, and interactive character.

What is this book about? It is about the AlgoRythmics universe. Of course, we have not dreamt up a complex teaching-learning tool and the attached didactical methods overnight. The AlgoRythmics project has its own particular history. Through this book, we invite the reader to accompany us as we virtually relive the AlgoRythmics adventure.

ISBN 978-606-975-044-5



9 786069 750445