Jean-Philippe Pellet
Gabriel Parriaux (Eds.)

Informatics in Schools

Beyond Bits and Bytes:
Nurturing Informatics Intelligence in Education

**16th International Conference on Informatics in Schools:
Situation, Evolution, and Perspectives, ISSEP 2023
Lausanne, Switzerland, October 23–25, 2023, Proceedings**

Springer

OPEN ACCESS

Lecture Notes in Computer Science 14296

The series Lecture Notes in Computer Science (LNCS), including its subseries Lecture Notes in Artificial Intelligence (LNAI) and Lecture Notes in Bioinformatics (LNBI), has established itself as a medium for the publication of new developments in computer science and information technology research, teaching, and education.

LNCS enjoys close cooperation with the computer science R & D community, the series counts many renowned academics among its volume editors and paper authors, and collaborates with prestigious societies. Its mission is to serve this international community by providing an invaluable service, mainly focused on the publication of conference and workshop proceedings and postproceedings. LNCS commenced publication in 1973.

Jean-Philippe Pellet · Gabriel Parriaux
Editors

Informatics in Schools

Beyond Bits and Bytes:
Nurturing Informatics Intelligence in Education

16th International Conference on Informatics in Schools:
Situation, Evolution, and Perspectives, ISSEP 2023
Lausanne, Switzerland, October 23–25, 2023
Proceedings

Springer

Editors
Jean-Philippe Pellet ⓘ
University of Teacher Education
Lausanne, Switzerland

Gabriel Parriaux ⓘ
University of Teacher Education
Lausanne, Switzerland

Preface

This volume contains full papers presented at the 16th International Conference on Informatics in Schools: Situation, Evolution and Perspectives (ISSEP 2023), which was held at the University of Teacher Education (HEP Vaud) in Lausanne, Switzerland, from October 23 to 25, 2023.

The ISSEP conference series is a forum for researchers and practitioners in the area of informatics education, in both primary and secondary schools. The conference provides an opportunity for educators and researchers to reflect upon the goals and objectives of this subject matter, its curricula, various teaching and learning paradigms and topics, as well as its connections to everyday life—including the various ways of developing informatics education in schools.

The conference series started in 2005 in Klagenfurt, Austria. Initially planned as a one-time event, interest was such that subsequent editions were organized in thirteen different countries to this day: in Vilnius, Lithuania (2006); Torun, Poland (2008); Zürich, Switzerland (2010); Bratislava, Slovakia (2011); Oldenburg, Germany (2013); Istanbul, Turkey (2014); Ljubljana, Slovenia (2015); Münster, Germany (2016); Helsinki, Finland (2017); St. Petersburg, Russia (2018); Larnaca, Cyprus (2019); Tallinn, Estonia (2020); Nijmegen, the Netherlands (2021); and Vienna, Austria (2022).

As with the previous edition in Vienna, a doctoral consortium was organized for ISSEP 2023, which received 13 applications from Ph.D. students. The doctoral consortium is a place for Ph.D. students to present and discuss their research ideas, meet each other as well as other senior researchers, and get constructive feedback from peers and researchers prior to the conference itself. This year's doctoral consortium was held on October 22, 2023, and was chaired by Engin Bumbacher from HEP Vaud.

On the first day of the conference, local teachers were also invited to attend. Practical workshops were organized on that day in addition to presentations. We believe closer interactions between teachers and researchers are important, on the one hand, in order to ensure that research is relevant to the classroom, and, on the other hand, to provide teachers with a view on the latest developments in the field. The Swiss Society for Informatics in Education (SVIA-SSIE-SSII), mainly composed of computer science teachers, supported the event and also organized its annual general assembly on the first day of the conference in Lausanne.

The ISSEP 2023 program committee received 73 submissions in total (including poster and workshop proposals), out of which 47 were paper submissions. Each of these was reviewed by between 3 and 5 members of the program committee. In total, 171 double-blind reviews were provided. We are extremely thankful for the dedicated and timely work of the reviewers! Based on the ratings and comments, 14 full papers were selected for publications in these proceedings, which corresponds to an acceptance rate of about 29.8%. The submission, review, and selection process was managed using the EasyChair conference management system.

The selected papers are organized into the following sections:

1. Artificial Intelligence and its Applications
2. Competitions, Problem Solving, and Computational Thinking
3. Robotics and Unplugged Modalities
4. Curricula & Computer Science Concepts

All sections contain a mix of research papers and experience reports/best-practice papers.

Included in this volume are also the abstracts of the talks given by the invited speakers: Lauren Margulieux, from Georgia State University; Shuchi Grover, from Looking Glass Ventures; and Helmut Schauer, retired from the University of Zurich.

Once again, we would like to thank the members of the Program Committee for the work they have done in reviewing the submissions and providing feedback to the authors. We thank the authors for their numerous, high-quality submissions. We are also very grateful to the members of ISSEP's steering committee for their advice and support. Andreas Bollin and Gerald Futschek, as organizers of the previous edition of ISSEP, provided particularly helpful guidance on a whole range of organizational matters and deserve special thanks. Finally, we thank our colleagues and members of the local organizing committee for their work in the concrete organization of the physical conference, as well as our institution's Grants Office, which helped us look for funding and provided support for the publication of these proceedings.

August 2023   Jean-Philippe Pellet
                                                         Gabriel Parriaux

Organization

Conference Chairs

Gabriel Parriaux	University of Teacher Education, Lausanne, Switzerland
Jean-Philippe Pellet	University of Teacher Education, Lausanne, Switzerland

Steering Committee

Erik Barendsen	Radboud University and Open University, The Netherlands
Andreas Bollin	University of Klagenfurt, Austria
Valentina Dagienė	Vilnius University, Lithuania
Gerald Futschek	TU Wien, Austria
Yasemin Gülbahar	Ankara University, Turkey
Juraj Hromkovič	ETH Zurich, Switzerland
Ivan Kalaš	Comenius University, Slovakia
Mart Laanpere	Tallinn University, Estonia
Sergei Pozdniakov	St. Petersburg Electrotechnical University, Russia

Program Committee

Andreas Bollin	University of Klagenfurt, Austria
Andrej Brodnik	University of Ljubljana, Slovenia
Julien Bugmann	University of Teacher Education, Lausanne, Switzerland
Engin Bumbacher	University of Teacher Education, Lausanne, Switzerland
Špela Cerar	University of Ljubljana, Slovenia
Morgane Chevalier	University of Teacher Education, Lausanne, Switzerland
Christian Datzko	Wirtschaftsgymnasium und Wirtschaftsmittelschule Basel, Switzerland
Monica Divitini	Norwegian Univ. of Science and Technology, Norway

Gerald Futschek	TU Wien, Austria
Micha Hersch	University of Teacher Education, Lausanne, Switzerland
Ivan Kalaš	Comenius University, Slovakia
Kaido Kikkas	Tallinn University of Technology, Estonia
Dennis Komm	ETH Zurich, Switzerland
Mart Laanpere	Tallinn University, Estonia
Martina Landman	TU Wien, Austria
Peter Larsson	University of Turku, Finland
Olivier Lévêque	Swiss Institute of Technology, Lausanne, Switzerland
Nina Lobnig	University of Klagenfurt, Austria
Birgy Lorenz	Tallinn University of Technology, Estonia
Maia Lust	Tallinn University, Estonia
Kati Mäkitalo	University of Oulu, Finland
Tilman Michaeli	TU Munich, Germany
Mattia Monga	Università degli Studi di Milano, Italy
Stefan Pasterk	University of Klagenfurt, Austria
Biljana Petreska von Ritter	University of Teacher Education, Lausanne, Switzerland
Hans Põldoja	Tallinn University, Estonia
Sergei Pozdniakov	St. Petersburg Electrotechnical University, Russia
Ralf Romeike	Freie Universität Berlin, Germany
Giovanni Serafini	ETH Zurich, Switzerland
Eva Schmidthaler	Johannes Kepler Universität Linz, Austria
Vipul Shah	ACM India CSpathshala Initiative, India
Gabrielė Stupurienė	Vilnius University, Lithuania
Reelika Suviste	University of Tartu, Estonia
Maciej Sysło	Warsaw School of Computer Science, Poland
Svetlana Unković	TU Wien, Austria
Patrick Wang	University of Teacher Education, Lausanne, Switzerland
Michael Weigend	University of Münster, Germany
Markus Wieser	University of Klagenfurt, Austria

Additional Reviewers

Walter Gander	ETH Zurich, Switzerland
Tobias Kohn	TU Wien, Austria
Alexandra Maximova	ETH Zurich, Switzerland

Doctoral Consortium Committee

Engin Bumbacher	University of Teacher Education, Lausanne, Switzerland
Andreas Bollin	University of Klagenfurt, Austria
Valentina Dagienė	Vilnius University, Lithuania
Gerald Futschek	TU Wien, Austria
Violetta Lonati	Università degli Studi di Milano, Italy
Tilman Michaeli	TU Munich, Germany
Jean-Philippe Pellet	University of Teacher Education, Lausanne, Switzerland

Local Organizing Committee

|	
Gabriel Parriaux	University of Teacher Education, Lausanne, Switzerland
Jean-Philippe Pellet	University of Teacher Education, Lausanne, Switzerland
Engin Bumbacher	University of Teacher Education, Lausanne, Switzerland
Biljana Petreska von Ritter	University of Teacher Education, Lausanne, Switzerland
Morgane Chevalier	University of Teacher Education, Lausanne, Switzerland
Julien Bugmann	University of Teacher Education, Lausanne, Switzerland
Patrick Wang	University of Teacher Education, Lausanne, Switzerland
Claire Matti	University of Teacher Education, Lausanne, Switzerland
Catherine Audrin	University of Teacher Education, Lausanne, Switzerland
Giovanni Serafini	ETH Zurich, Switzerland

Invited Talks

Computing Across the Curriculum: CS Knowledge and Skills That Everyone Values

Lauren Margulieux

Department of Learning Sciences, Georgia State University

This talk will explore how computing can be used to support teaching and learning across the curriculum, especially in non-computer-science classes. It will focus on identifying computing concepts that are most relevant in non-computer-science domains and, thus, most useful across the curriculum and important for general computational literacy. To identify relevant concepts and provide examples of computing across the curriculum, the talk will first explore five years of work at Georgia State University to co-design integrated computing activities with non-computer-science teacher education faculty and prepare future teachers to use the activities.

To complement this qualitative, design-based approach, the talk will also present quantitative, summative analyses of computing concepts that are taught in integrated computing activities. Activities and curricula were collected from around the globe to examine how programming and other computing concepts and practices are used for activities in non-computer-science classrooms. The talk will focus on the emergent paradigms for integrated computing activities, common computing concepts already used, opportunities for expanding computing tools, and how well these activities prepare students for later, standalone computing and programming courses.

Based on these analyses, recommendations for teacher preparation and integrated computing activities in primary and secondary schools will be made. In addition, lessons learned about strategies for increasing teacher buy-in and coherence with current education practices will be discussed. The goal of the talk is both to improve current practice based on emerging data about integrated computing activities and to identify areas of opportunities for growth related to teachers' and students' computer science knowledge and skills to better support teaching, learning, and computational literacy.

Lauren Margulieux is an associate professor at Georgia State University in the Department of Learning Sciences. She is passionate about helping others to develop skills and pursue opportunities. In particular, she focuses on spreading computational literacy and the use of computing to achieve personal and professional goals.

Teaching AI in K-12: Examples, Issues and Guidance from K-12 CS Education Research

Shuchi Grover

Looking Glass Ventures

"By 2024, more than eight billion AI-powered digital voice assistants (a number roughly equal to the world's population) will be in use globally."
(Thormundsson, 2022).

There is growing recognition of the need to teach about artificial intelligence and machine learning (AI/ML) at the school level in light of the meteoric growth in the range and diversity of applications of machine learning (ML) in all industries and everyday consumer products, with Large Language Models (LLMs) being only the latest and most compelling example yet! Efforts to bring AI, especially ML education, to school learners are being propelled by substantial industry interest and efforts such as AI4K12, as well as technological developments that make AI tools readily available to learners of all ages. These efforts span a variety of learning goals as well as pedagogies ranging from exploratory, playful interactions with pre-trained AI models, the extension of K-12 introductory CS activities to include AI tools such as classifiers, integration of AI into other subjects, activities that lift the hood on how AI works, critical examination of ethical and societal issues that are exacerbated by bias in algorithms, and unplugged activities that make complex AI/ML ideas accessible to younger learners. What are the emerging lessons from early AI education research efforts? What challenges and issues do K-12 curriculum designers need to address in designing for teaching AI in K-12? How should teachers be prepared to teach this novel subject? What are key lessons from K-12 CS research and practice efforts that can provide guidance on how to purposefully address the pertinent, topical question of how to teach AI in school and tackle what to many feels like "the next new thing"?

In this keynote address, Shuchi Grover will share examples from the field as well as her own research into designing for AI learning in high schools—designing AI/ML curricular modules that focus on socially relevant applications, and integrating AI learning into high school cybersecurity curricula. She will also draw on her deep expertise developed over 15 years in K-12 CS education research to highlight key lessons from two decades of CS education research and practice that can help build on successes while mitigating missteps in K-12 AI Education.

Shuchi Grover is a computer scientist and learning scientist by training. She has been committed to PK-12 computer science education in formal and informal settings for over two decades. Formerly a senior researcher at SRI International's Center for Technology in Learning and Visiting Scholar at Stanford University, she is currently senior research scientist at Looking Glass Ventures where she leads several NSF-funded

projects involving research & design of curriculum, assessments, tools, and environments that help develop twenty-first century competencies in topics such as computing, STEM+CS integration, data science, AI, and cybersecurity as well as issues of neurodiversity, gender equity, and teacher preparation. She created, co-authored and edited Computer Science in K-12: A-to-Z Handbook on Teaching Programming.

Shuchi has a Ph.D. in Learning Sciences & Technology Design with a focus on K-12 CS Education (Stanford University), master's degrees in education (Harvard University) and computer science (CWRU, Cleveland), and bachelor's degrees in computer science and physics (BITS Pilani, India).

Informatics in Schools and Everyday Life

Helmut Schauer

Professor Emeritus at the University of Zurich

Because of the short-term usability of product specific skills, we have to focus on conceptual knowledge to guarantee useful long-lasting educational benefits for pupils and teachers as well. Examples of long-term concepts include topics like "notions and notations", "information, codes and redundancy", "significance and plausibility", "modelling and abstraction", "formalized systems", "determinism versus chaos", "orders of magnitude" and "algorithmic complexity".

These concepts are illustrated by everyday life situations like "optimizing" the loading of a dish-washer or the order of cookies on a baking tray, "stacks and queues" in supermarkets, "strategies" to solve a puzzle, advantages of "simulating" weightings with a beam balance or the "digitalization" of human characteristics. The use of computers obviously constructs realities with all the benefits and drawbacks. Let the wisdom gathered from the fruits of the "tree of knowledge" lead us to ensure desirable lives for us and our successors.

Helmut Schauer was a full professor at the Department of Informatics of the University of Zurich. He was the head of the Educational Engineering Research Group. His research interests include web-based and game-based learning, assessments beyond multiple choice, collaborative learning environments, object-oriented programming in Java and visualization of algorithms and data structures.

He has contributed to numerous discussions on curriculum issues at various levels of education. His special interest focuses on curricula and didactics of informatics in secondary school levels. He is a past president of the Swiss Informatics Society SI and a board member of ECDL-SI, which oversees the operations of the ECDL Program in Switzerland.

Contents

Artificial Intelligence and Its Applications

Education and Awareness for Artificial Intelligence

Martin Kandlhofer[1]([✉])[iD], Petra Weixelbraun[2][iD], Manuel Menzinger[3][iD], Gerald Steinbauer-Wagner[4][iD], and Ágoston Kemenesi[5][iD]

[1] Austrian Computer Society OCG, Vienna, Austria
martin.kandlhofer@ocg.at
[2] University of Vienna, Vienna, Austria
weixelbraun@uni-ak.ac.at
[3] KLEX-Klusemann Extern, Graz, Austria
menzinger.manuel@klex.co.at
[4] Graz University of Technology, Graz, Austria
steinbauer@ist.tugraz.at
[5] Mobilis Interactive Exhibition Center, Győr, Hungary
kemenesi.agoston@mobilis-gyor.hu

Abstract. The increasing digitization and automation processes in daily life through the use of Artificial Intelligence (AI) pose great challenges for society and education. These range from building awareness, increasing acceptance, and teaching the foundations of this important and disruptive technology, to fostering a meaningful, creative usage, an assessment of threats, opportunities, and potentials as well as allowing an informed discussion about the technology. This paper presents the 2-year international AI education and awareness project 'ENARIS' which addressed these challenges on various levels. On the one hand, the project fostered young people's interest in AI and facilitated a basic technical understanding. In this context, the integration of teachers, using a train-the-trainer approach and providing ready-to-use, open educational resources based on sound didactic concepts was an essential factor. On the other hand, the project aimed at strengthening awareness regarding social, economic, and technical aspects and potentials of AI among the general public, including school students, children, parents or working persons by conducting open and easily-accessible workshops. In the first project stage, online pre-surveys were conducted to analyze the needs within the target groups. Based on the results, AI ready-to-use prototype learning modules were developed. Following the principles of constructionism, a combination of different teaching methods including unplugged and plugged activities was used. The second project stage dealt with implementing and evaluating pilot workshops using quantitative pre- and post-tests as well as qualitative measures. Results indicate that, a) the ready-to-use teaching material, train-the-trainer workshops and AI topics covered were well received and that, b) a significant positive impact regarding the awareness and general knowledge about AI was achieved.

1   Introduction

Artificial Intelligence (AI) is already part of our daily life and the working world. To ensure a sustainable and responsible usage of this disruptive technology, young people with a sufficient understanding of AI and skills for using these new technologies are required. Stimulating enthusiasm as well as facilitating a basic understanding has to be done at an early age in order to foster AI literacy. According to Long & Magerko AI literacy can be defined as *"a set of competencies that enables individuals to critically evaluate AI technologies; communicate and collaborate effectively with AI; and use AI as a tool online, at home, and in the workplace"* [14]. Fostering AI literacy goes hand in hand with fostering awareness and general knowledge about AI, providing a sound basis for young peoples' decision to pursue a career in an AI-related sector and enabling social and economic participation.

This paper presents methods and results of the international project 'ENARIS' *(Education and Awareness for Intelligent Systems)* which aimed to foster AI awareness and a general understanding of AI concepts. The project lasted two years, 472 teachers and young people were trained and 73 workshops for teachers, school students and the public were held and empirically evaluated (pre-survey among target groups, pilot implementations, stakeholder reviews, pre-/post-tests of workshops). In order to ensure versatile access to the thematic blocks, researchers from the fields of computer science as well as the humanities with a focus on ethics and art were involved in the development of the content.

The remainder of the papers is structured as follows: Sect. 2 provides an overview of related literature, Sect. 3 discusses the applied didactic methodology and also provides an overview of the AI learning modules developed. Section 5 presents evaluation methods and results, while conclusions, limitations and future work are discussed in Sect. 6.

2   Related Work

Traditionally, teaching AI concepts has mostly been done at the university level. Nevertheless, in recent years AI education at K-12 level has become a major topic and is still evolving [12]. For instance, the *AI4K12* [2] initiative focuses on the development of AI education guidelines and centers its concepts around the *Five Big Ideas in AI* (perception, representation and reasoning, learning, natural interaction, societal impact) [22]. Additionally, an online repository provides supporting material for teaching AI at K-12 level. The initiative *Elements of AI* [8] provides a free e-learning course, covering foundations of AI in an easy comprehensible form and targeting a general audience. An example for fostering AI skills through unplugged activities is the project *AI Unplugged* [18]. It provides a collection of paper-and-pencil activities to teach decision trees, deep

learning, reinforcement learning, problem-solving using search and the Turing Test. In her paper, Kasinidou presents an ongoing project which investigates how people perceive and comprehend AI across various segments of the public, including children and adults [13].

An extensive discussion of further existing AI K-12 initiatives and projects along four dimensions (formal/informal education, cooperation between AI and education research and teachers, level of AI education - from broad to specific, concepts and tools for teaching AI to youngsters) can be found in the article by Steinbauer, et al. [20]. The study by Casal-Otero, et al. [5] provides an overview of how AI is currently integrated into K-12 education. In this context, Tenório, et al. also conducted a bibliometric analysis, investigating the publications in the area of AI literacy from 1989 to 2021 [21]. The article by Olari & Romeike analyzes the correlation between AI and data literacy skills within current educational frameworks [16].

Compared to already existing projects and initiatives, the project presented in this paper follows a hybrid learning approach with a strong focus on educators (teachers, trainers) which acted as multipliers later on. Within project duration, ready-to-use, freely available teaching material was developed (open educational resources). In addition, in-person (face-to-face) and virtual train-the-trainer courses and workshops for young people were conducted. Finally, the project not only addressed teachers and educators, but also the broad general public by developing and conducting workshops 'for everybody'.

3 Methodology

3.1 Pre-survey

In the beginning of the project, a needs analysis (pre-survey), in the form of two separate online questionnaires, one for teachers and one for the general public, was conducted. The survey was divided into three sections corresponding to AI topics and concepts, teaching material as well as personal information. In sum, the pre-survey comprised 65 Likert scale questions and five free text questions to allow remarks and personal opinions. The survey questions were written in English and translated into German and Hungarian (the official languages of the two project countries). The surveys were then distributed using *LimeSurvey* and *Google Forms*. The teacher survey was sent to school educators in Austria and Hungary, to which 143 teachers replied. The survey focusing on the general public was conducted mostly in a science center in Hungary and comprised 82 responses. The needs analysis revealed that the most interesting type of learning material to use are short and independent thematic units which include interactive elements like tutorials and simulations in combination with age-appropriate explanations of technical concepts. Participants highlighted their strong interest in the topics *social impact of AI* and *machine learning*. Furthermore, the results indicate a general interest in the AI topic. In this context, 52% of the participants of the teacher survey stated that they are currently teaching AI in school or are

planning to do so in the future[1]. In a further step, it was therefore also necessary to consider how the interest of the other 48% could be raised in order to teach AI related topics in the classroom.

3.2   Didactic Methods

Based on these results, teaching material in a modular form was created. The aim was to help teachers to incorporate AI related topics into their classes and to design and implement workshops to foster a more basic understanding for the technology as well as to strengthen awareness for AI. In order to ensure versatile access to the thematic blocks and consequently also ensure interest of a higher number of teachers from different subjects, researchers from the fields of computer science as well as the humanities with a focus on ethics and art were involved in the development of the content. This interdisciplinary approach during development is also intended to reflect and promote cross-disciplinary collaboration in different subjects and school project work. In addition, through these multifaceted approaches and enclosed detailed theoretical information materials, uncertain teachers who lacked the necessary know-how or connection possibilities in their subject should be introduced to the topic with a low entry bar. In order to interest the greatest possible number of students in the topic, to initiate learning processes in the sense of a participatory fairness and to open up inclusive learning spaces, the majority of the materials were designed in a differentiated way and opportunities for individualized access were created. Based on the differentiation according to Finkelstein, Sharma, and Furlonger (2019), the learning objectives are basically the same for all students, but collaborative exercises, additional materials in different media variants, and different degrees of complexity in the questions and tasks are provided [9]. In addition, the use of digital as well as unplugged exercises should enable these materials to be used independently of the school's equipment [4]. To provide an immersive learning experience, improve the learning efficiency and motivation, gamifying elements such as scoring, competition and storytelling and the resulting slipping into roles were used [19]. In addition to this approach of using constructivist principles [17], there has also been an emphasis on the 21st century skills [10], especially on the 4Cs, creativity, critical thinking, collaboration, and communication so that students should be given the opportunity to share their thoughts, questions, ideas and solutions on different topics concerning AI.

4   Implementation

4.1   Learning Modules

Initially, three prototype modules were created, building on the findings of the pre-survey (see Sect. 3.1). These modules focused on the topics *AI Basic*, *Ethics* and *Supervised Learning*. The prototype modules were qualitatively evaluated

---

[1] Further details regarding the survey and the results are available upon request.

within the scope of a pilot review workshop with teachers (further details, see Sect. 5).

In general, each module consists of a structured lesson plan containing content for two to four hours and includes hands-on activities as well as material for theoretical input. Furthermore, a written summary of the subject, in the form of a teacher guide, is included to provide the required knowledge as well as references for further research. The following paragraph provides an overview of the different modules.

AI Basics. While all modules are mostly independent of each other, there is some knowledge that relates to most modules. This module includes a basic definition of AI as well as common terminology like algorithms and data, as well as an overview of the vast field of AI. Therefore, this module acts as a natural starting point and ensures that other modules do not need to reconsider these basics. In addition, this module is designed for a duration of only one to two hours. Thus, it can be covered before any others easily.

Ethics. In this module, the students learn a critical approach to AI and the need for ethical guidelines. Among other things, they learn to formulate and test their own robot laws, reflect on their own and common viewpoints on data bias, and transfer thought experiments, such as the trolley problem, to the current difficulties of autonomous driving.

Supervised Learning. In addition to the basics of what a supervised learning model is and the meaning of training by using data, this module includes chapters about overfitting and underfitting as well as possibilities and limitations of supervised learning algorithms. On the practical side, students have to create their own model to differentiate between pictures of cats and dogs. Finally, they have to train a real model to recognize directions and use this final model to control a game of snake.

Chatbots - Natural Language Processing. In this module, different types of chatbots are illustrated through various practical, gamified exercises. The students learn what chatbots are and how they work. The authentic imitation of human speech, associated problems and the Turing Test are addressed.

Reinforcement Learning. In this module, basic content regarding reinforcement learning (RL) is covered in the form of small educational games. In addition, students learn about the basic RL-process and decision making based on Q-values.

Computer Vision. Here, the most striking differences between human and machine visual perception are identified. It is intended to elaborate on how a computer stores and processes visual information and where the limits and possibilities lie. The students work with practical hands-on examples and develop a face recognition algorithm using the visual programming language *Scratch*.

Neural Networks. In this slightly more challenging module, students learn the basic structure and operation of neural networks and how they can be used practically by testing simulations from the module materials.

Art and Artificial Intelligence. This module deals less with the technical functioning of image-generating applications, but focuses more on the ethical

and socially relevant issues. Students discuss copyright issues, definitions of creativity and art, and whether an AI is even capable of making art. In this context, students test easily accessible image generation applications.

AI and Manipulation in Social Media. The topic of manipulation by AI is examined from two sides in these materials. On the one hand, the handling of one's own private data, which are used by companies to generate customized advertising, is discussed by evaluating the student's own data on social media by themselves. On the other hand, the topic of deepfake is taken up, how they are developed and what damage they can cause. As a consolidation, deepfakes can be created with the help of commonly used applications.

AI and Environment. In project lessons, the students reflect on their individual energy consumption and detect digital "climate killers" in their everyday life and in the development of novel AI technologies. In a further step, students reflect on current environmental and climate problems caused through the use of technologies in the form of an ideas laboratory, where they sketch and discuss solutions based on AI.

All modules are published under the creative commons BY-SA license and can be found on the project website[2].

4.2   Regular Implementations

During the 2-year project duration, 472 persons (203 teachers and 269 school students) were trained, and 73 workshops were held. Figure 1 provides an overview of these regular implementations.
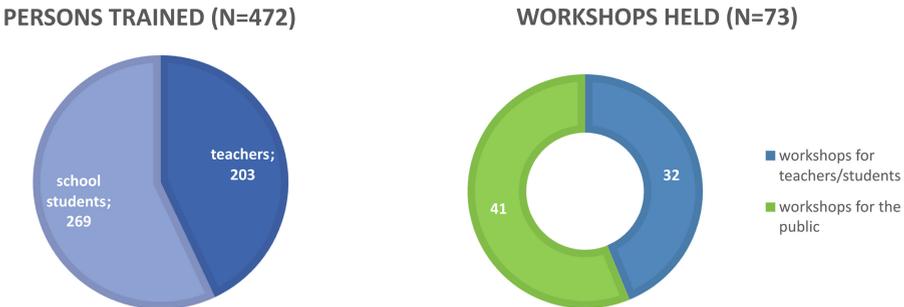

Fig. 1. Number of teachers and school students trained (left) and workshops held (right)

---

5   Evaluation Methods and Results

A combination of well-grounded quantitative and qualitative evaluation methods were used [7,15] throughout the project.

After conducting and analyzing the quantitative **pre-survey** (details and findings were described in Sect. 3.1) and developing three prototype learning modules (as discussed in Sect. 4.1), a **pilot review workshop** was held with four selected secondary school teachers, acting as external experts. The workshop started with a hands-on session, where teachers extensively tested the three prototype modules. Afterwards, each teacher was interviewed by the project researchers using a semi-structured interview approach [11]. Summarizing the results, the prototype modules were received positively. A specific critique addressed the way of providing the modules to the public. Therefore, the publication method was changed from offline documents to a website, with the option to download everything for offline use and printing.

Furthermore, two **stakeholder review workshops** with public education authorities and representatives from the industrial sector from each project country were conducted at the midway point of the project. In the workshops quantitative methods (i.e. World-Cafe, discussion) were applied to gather feedback from the stakeholders in order to ensure an alignment with educational and economic strategies as well as maximizing an adoption of the project outputs. In this context, the stakeholders specifically recommended matching the modules with the competence frameworks for national school curricula[3]. Based on the findings of the pilot workshop and the data from the stakeholder review workshops, the modules were adapted and served as templates for all subsequently developed modules.

In order to evaluate the impact on participants' awareness and general knowledge of AI, quantitative **pre- and post-tests** were conducted before and after a workshop. The instrument applied was a six-item true/false questionnaire based on the questionnaire used in the national survey called *America Needs AI Literacy Now* which was conducted in the US in 2021 with over 1500 participants [1,6]. Questions were translated from English to German and Hungarian (by native speakers) and covered the areas *artificial versus human intelligence*, *AI in everyday life* as well as *limitations and societal aspects of AI*[4].

Pre- and post-test data was collected during seven representative pilot-workshops for the general public (comprising young people as well as working persons), with in total 57 participants. In order to ease the participation (e.g. using smartphones) as well as to apply a gamification approach, the online tool *AhaSlides* was used to perform the tests [3]. No personal data (like age or gender) was collected, ensuring participants' anonymity. The data gathered was analyzed using inferential statistical procedures (paired-samples t-test) [7].

---

[3] As a first step, this recommendation was implemented in Austria for the curriculum 'Digital Basic Education'; see https://enaris.org/material/de/education.html.

[4] The six-item questionnaire will be provided upon request.

Compared to the results of the pre-tests (M = 52.83, SD = 22.88), participants showed a significant gain in AI awareness/knowledge after the workshops (post-tests; M = 74.85, SD = 21.62; t(56) = 7.29; p < .001). The effect size (measured by Cohen's d) was calculated with d = 0.97, indicating a large effect (see Fig. 2, right picture). An improvement could be observed for each of the six questions (as shown in Fig. 2, left picture). These results indicate that the workshops had a positive effect on participants' awareness and general knowledge of AI.
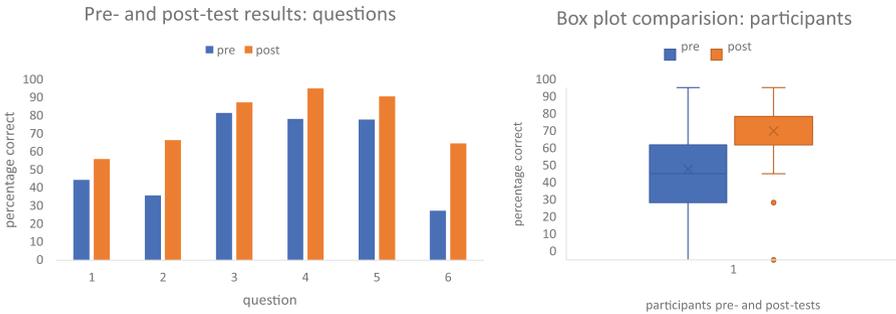
Fig. 2. Pre- and post-test results of the AI questionnaire (n = 57); analyzed by questions (left) and by participants (right)

6   Conclusion and Outlook

Artificial Intelligence (AI) has become increasingly important, having an influence on many aspects of our lives. The emergence of large language models (like ChatGPT) in the recent months brought AI into the spotlight, igniting curiosity and discussions worldwide. Recognizing the need to foster a basic technical understanding of AI and raise awareness about its implications, the AI education and awareness project 'ENARIS' presented in this paper was initiated, implemented and empirically evaluated (pre-survey among target groups, pilot implementations, stakeholder review workshops, pre-/post-tests). The results of these evaluations were encouraging, highlighting the significance of such activities and projects in enhancing AI awareness and general knowledge among young people, teachers and the general public. The quantitative data - though limited by a relatively small sample size - demonstrated a significant improvement of participants' awareness and general knowledge of AI. The qualitative data revealed a positive perception of the overall project concept and the developed learning modules. The next steps comprise the implementation of further workshops for trainers and teachers (acting as multipliers) as well as the further development of the learning modules (since new AI tools and applications literally emerge every week). Due to the open-source nature of the project, we also count on the active involvement of the community. Finally, we plan a more comprehensive evaluation, comprising a larger sample size as well as a more extensive AI knowledge test (which is currently being developed).

By providing open educational resources, based on sound didactic concepts, by applying a train-the-trainer approach, by integrating stakeholders in the review process, and finally, by conducting AI workshops for teachers, school students and the general public, we envision a sustainable positive impact of the project on the understanding and usage of AI as well as on fostering a more informed and AI-aware society.

References

1. ACM: America Needs AI Literacy Now (2021). https://cacm.acm.org/news/257309-america-needs-ai-literacy-now/fulltext. Accessed 26 May 2023
2. AI4K12: The Artificial Intelligence (AI) for K-12 Initiative (2023). https://ai4k12.org/. Accessed 26 May 2023
3. Ameen, S., Praharaj, S.K.: Interactive workshop on writing and publishing research: reflections and lessons learned. Kerala J. Psychiatry **34**(1), 21–26 (2021)
4. Bell, T., Vahrenhold, J.: CS unplugged—how is it used, and does it work? In: Böckenhauer, H.-J., Komm, D., Unger, W. (eds.) Adventures Between Lower Bounds and Higher Altitudes. LNCS, vol. 11011, pp. 497–521. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98355-4_29
5. Casal-Otero, L., Catala, A., Fernández-Morante, C., Taboada, M., Cebreiro, B., Barro, S.: AI literacy in K-12: a systematic literature review. Int. J. STEM Educ. **10**(1), 29 (2023)
6. DeCario, N.: America Needs AI Literacy Now (2021). https://blog.allenai.org/american-needs-ai-literacy-now-141b8cd17a83. Accessed 26 May 2023
7. Diekmann, A.: Empirische Sozialforschung. Anwendungen; Rowohlt, Grundlagen, Methoden (1995)
8. ElementsOfAI: Free Online Courses. University of Helsinki and Reaktor (2022). https://www.elementsofai.com/. Accessed 26 May 2023
9. Finkelstein, S., Sharma, U., Furlonger, B.: The inclusive practices of classroom teachers: a scoping review and thematic analysis. Int. J. Inclusive Educ. **25**(6), 735–762 (2019). https://doi.org/10.1080/13603116.2019.1572232
10. Geisinger, K.F.: 21st century skills: What are they and how do we assess them? Appl. Measur. Educ. **29**(4), 245–249 (2016)
11. Hove, S.E., Anda, B.: Experiences from conducting semi-structured interviews in empirical software engineering research. In: IEEE International Software Metrics Symposium (2005). https://doi.org/10.1109/METRICS.2005.24
12. Kandlhofer, M., et al.: EDLRIS: a European driving license for robots and intelligent systems. KI-Künstliche Intelligenz **35**, 221–232 (2021)
13. Kasinidou, M.: Promoting AI literacy for the public. In: Proceedings of the 54th ACM Technical Symposium on Computer Science Education, vol. 2, p. 1237 (2022)
14. Long, D., Magerko, B.: What is AI literacy? Competencies and design considerations. In: Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, pp. 1–16 (2020)
15. Morgan, D.L.: Practical strategies for combining qualitative and quantitative methods: Applications to health research. Qualitative Health Res. **8**(3), 362–376 (1998)

16. Olari, V., Romeike, R.: Addressing AI and Data Literacy in Teacher Education: A Review of Existing Educational Frameworks. In: The 16th Workshop in Primary and Secondary Computing Education. pp. 1–2 (2021)
17. Papert, S., Harel, I.: Situating constructionism. Constructionism **36**(2), 1–11 (1991)
18. Seegerer, S., Lindner, A.: AI Unplugged; Friedrich-Alexander-Universität Erlangen-Nürnberg (2022). https://www.aiunplugged.org/. Accessed 26 May 2023
19. Siemon, D., Grogorick, L.: Gamification of teaching in higher education. In: Gamification - Using Game Elements in Serious Context, pp. 153–164 (2016)
20. Steinbauer, G., Kandlhofer, M., Chklovski, T., Heintz, F., Koenig, S.: A differentiated discussion about AI education K-12. KI-Künstliche Intelligenz **35**(2) (2021)
21. Tenório, K., Olari, V., Chikobava, M., Romeike, R.: Artificial intelligence literacy research field: a bibliometric analysis from 1989 to 2021. In: Proceedings of the 54th ACM Technical Symposium on Computer Science Education, vol. 1, pp. 1083–1089 (2023)
22. Touretzky, D., Gardner-McCune, C., Martin, F., Seehorn, D.: Envisioning AI for K-12: what should every child know about AI. In: AAAI Conference on Artificial Intelligence (2019). https://doi.org/10.1609/aaai.v33i01.33019795

What Is AI-PACK? – Outline of AI Competencies for Teaching with DPACK

Uwe Lorenz[(✉)] and Ralf Romeike

Free University of Berlin, Königin-Luise-Str. 24-26, 14195 Berlin, Germany
{uwe.lorenz,ralf.romeike}@fu-berlin.de

Abstract. The changes brought by digital transformation have already been addressed in various places for schools and teacher training. Among other things, competence frameworks have been developed for practical orientation and to describe required skills. However, the reflective handling of the now widespread phenomena of the "AI world" requires different competencies than the handling of conventional informatics systems mainly discussed so far, because "AI systems" are based on a different design approach that gives these systems special properties. This, in turn, poses new demands on the professionalization of all teachers, since the development has clear implications for their central areas of action. The goal of this paper is to define a framework that outlines these new AI-related requirements to support the design of holistic education and study programs for teaching in the digitally networked "AI world" that give equal weight to the user-oriented, technological, sociocultural perspective. This contribution approaches the topic theoretically: first, the need to delineate an area of AI-related digital competencies is justified. Then this area is located in the DPACK model. Since teachers combine pedagogical, subject-related, and technical competencies in their work, the domains of "AI related Pedagogical Knowledge" (AI-PK), "AI related Content Knowledge" (AI-CK), and "AI related Pedagogical Content Knowledge" (AI-PCK) are deductively determined and illustrated with exemplary competencies. The "AI-PACK" framework enables a structured description and investigation of AI teacher education requirements.

Keywords: DPACK · AI-PACK · Artificial Intelligence · competence model · teacher training

1 Introduction

The digital transformation continues to progress dynamically. The changes brought by the ICT are influencing how we communicate, work or gather information in all areas of our daily life. A new stage of development is reached by the technologies of "Artificial Intelligence" (AI). Due to the given technological prerequisites, a rapidly increasing spread can be observed. On the one hand, over-arching fields of action are affected - media education, advisory and assessment

tasks change or become superfluous - on the other hand, a wide range of new opportunities arise for subject-specific teaching, but also challenges with regard to forms, content and methods. Due to the given technical conditions, a rapidly increasing diffusion of these technologies can be observed. This again places considerable demands on the professionalization of teachers in all subjects. On the one hand, overarching fields of action are affected - media education, advisory, and assessment tasks change or become superfluous - on the other hand, a wide range of new opportunities arise for subject teaching but also challenges with regard to its forms, content, and methods.

With regard to the interdisciplinary cross-sectional task of supporting the design of holistic educational and study programs for teaching in the digitally networked "AI world", the following questions arise: 1. What are the special features of AI systems compared to conventional informatics systems, as they have mainly been discussed in the context of digital transformation so far? 2. What does the AI-related area of competencies for teachers encompass? Since teachers combine pedagogical, subject-specific and digitality related competencies in their work, general "AI competencies" are not sufficient. Following the Sect. 3, in which, among other things, the special features of AI systems are discussed, the teaching-related AI competence areas are therefore characterized deductively in the Sect. 4 on the basis of the DPACK model and illustrated with some examples.

2 Related Work

The profound changes brought by the digital transformation have been addressed in school and teacher education in numerous works and competence frameworks for structured description and exploration as well as for practice orientation have been created, such as the US "Framework for 21st Century Learning" [17]. It was developed by the non-profit organization "Partnership for 21st Century Learning" (P21), which is made up of representatives from industry, education, and the public sector and addresses required skills in the domains of "learning & innovation", "information, media & technology" and "life & career". Another framework that aims to offer an understanding of what "digital competence" is, is the EU framework "DigComp". This is not in the context of teacher education, but describes competencies that citizens need to live in a "digital world". It addresses the areas "Information and data literacy", "Communication and collaboration", "Digital content creation", "Safety" and "Problem-solving". The newer version "DigComp 2.2" [4] already includes an AI-related update. The European Framework for the Digital Competence of Educators "DigCompEdu" [3] builds on an older version 2.0 (2016) of the EU digital literacy framework DigComp, which did not yet include an AI update. A widely accepted model for teacher competencies is TPACK [14]. TPACK and the DPACK [8] based on it (see Sect. 3.1), have origins in Computer Science Education (CSE). In our consideration, we want to take into account not only user-oriented but also technological and socio-cultural perspectives according to the "Dagstuhl Declaration" (cf. [5]) where it is considered on an equal footing, presented as three

sides of the so-called "Dagstuhl Triangle". What is explicitly taken into account in DPACK. With its focus on digital competencies and its background in CSE, DPACK is well suited for our interdisciplinary cross-cutting task, which we want to address, namely to support the design of holistic education and study programs for teaching in the digitally networked "AI world".

In the scientific literature where TPACK or its derivatives appear in the context of "AI", there are several works that use TPACK to describe and explore teaching and learning that has AI as its subject like Seonghun Kim et al. [13], Druga et al. [6], and [20]. However, contributions that, like this one, conversely aim to determine the required AI competencies to teach have been scarce. In [25], under the impression of the digitization push in the context of Covid-19, there are some general indications of what the inclusion of AI in teaching methods and contents as well as in the design of teaching-learning environments could mean, but socio-cultural and technological perspectives, which we want to address in our approach here, are not further considered there. Celik [2] describes an AI competency model derived from TPACK, which also includes an ethical component. He mainly has a specific subset of AI systems in mind here, as the work focuses heavily on a field that targets specific application competencies of AI-based self-learning tools, i.e., tools that provide individualized, adaptive feedback in real-time, with ways for the teacher to analyze learning progress, etc. However, the field of possible applications of AI software is much wider, as shown in [10]. Also, in comparison, "digital competence" is understood in a more holistic way in the DPACK model. Scientific contributions that, like the present one, aim to determine required AI-related competencies for teachers holistically, taking into account the Dagstuhl perspectives, do not seem to exist so far.

Moreover, there is no consensus with regard to a definition of those informatics systems that produce the AI phenomena addressed (cf. Sect. 3.2). For an overview of this, the European Commission's "AI Watch" report [18] provides an informative source. Here, in the context of a possible political and legal evaluation, 64 AI definitions and provisions from politics, industry, and research are compiled and evaluated. Other recent articles describing the basic characteristics of AI, reflecting the state of the debate in terms of societal, cultural, or ethical challenges, and presenting new potential applications in education include. [10,15] (Sect. 3.3). A competency framework with a CSE background for K-12 education from which possible and necessary AI competencies can be obtained was presented with [16]. [21] discusses from a CSE perspective some fundamental shifts regarding "AI Thinking" or "Computational Thinking 2.0" compared to the "Computational Thinking" discussed so far, especially seen in the context of "Machine Learning" systems. To justify our framework, in Sect. 3.2 a provision that we consider appropriate is given and presented for discussion.

3 Theoretical Background

In this section, the structure and background of TPACK and DPACK are first briefly presented. It then justifies the need to consider separately the "AI-K"

domain of AI-related competencies (the "D" of DK has been replaced here by "AI") within the "digital literacy" (DK) domain by addressing the special characteristics of AI and showing that problems and requirements need to be taken into account here that do not occur in conventional computer science systems.

3.1   The TPACK and DPACK Models

In the TPACK competency framework the three domains of teacher professional knowledge introduced by Shulman [19], general "Pedagogical Knowledge" (PK), subject-matter "Content Knowledge" (CK), and the domain of "Pedagogical Content Knowledge" (PCK), are complemented by a domain of "Technological Knowledge" (TK) (cf. [14]). Replacing the "T" with a "D" at DPACK [9] is intended to emphasize that not only technical application knowledge but a "Digitality related knowledge" is taken into account in the TK sector, which is in DPACK characterized by the three perspectives of the Dagstuhl Triangle (Fig. 1). "Digitality related knowledge" (DK) is the necessary competence to be able to recognize, describe, reflect, and shape phenomena in a culture of digitality [8]. The use of term competence is also intended to illustrate that the requirements for teachers are not only at the level of "knowledge" [8].If it were solely about discussing and explaining digital literacy in the context of phenomena of the "digital networked world," it would be sufficient to apply the Dagstuhl model as in [16]. However, teachers' digital competence must additionally be discussed in the context of their content-related (i.e., subject-related) and pedagogical competencies (cf. [8] Area D).

3.2   AI Systems as Special Informatics Systems

The AI-PACK competence framework refers to a subdomain of informatics phenomena. Informatics phenomena are events caused by automated information processing. In order to define this domain, appropriate identification of the informatics systems that produce these phenomena is needed. Here, some difficulties arise initially in the context of AI. The AI Watch" [18] report notes that AI is usually described in relation to human intelligence or intelligence in general, with many definitions referring to machines that behave like humans or are capable of actions that require intelligence. Consequently, these definitions are ambiguous or describe a "moving target", as in "Tesler's Theorem" ("AI is whatever hasn't been done yet.") or generate undesirable anthropomorphic associations. Moreover, AI need not behave human-like in any way; for example, typical non-AI tasks" such as mathematical calculations can also be performed by products of AI techniques such as Machine Learning. Definitions of a technical perspective ("How does it work?", "How was it made?") list specific techniques and approaches used to develop appropriate software, such as the European Commission's "AI Act" definition [18]. This results in clearer provisions, but such a list must be permanently updated, especially if it is very detailed and does not refer to central principles.

Therefore, the question arises: what specifically characterizes automated information processing in AI systems? A representation of a CS design approach in AI results from considering the usual division of the field into 1. "knowledge-based AI", sometimes also called "classical", "symbolic" or "rule-based AI" (GOFAI, "Good Old-Fashioned AI") and 2. "Machine Learning" (ML) [10,16]. In GOFAI systems, a "knowledge base" is built using appropriate structured and prepared data representing content ("facts") that form the basis for a heuristic and rule-based search for precisely specified solutions. ML systems follow a different paradigm in that the initial search is not for solutions, but in the space of possible functions [12]. In ML, functions are found through an iterative, data-driven optimization ("training process"), usually with the help of examples and a complex approximation procedure that includes an objective function [10]. A function found with it is finally used in the application context for the computation of usable solutions. That the developed software follows comprehensible rules, is thereby no condition. Only statistical tests take place for the examination. This basic principle can be found in the different variants of ML [11]. Here we refer to ML software the software that performs such optimization, as well as the software that is the product of such a process.

In summary, then, we follow here a view according to which an alternative problem-solving paradigm is applied in AI systems, in which the process of the information processing that is to produce the desired outputs need not be described. Historically central to this approach are: first, the search for answers in a knowledge base using general inference rules (GOFAI), and second, the data-driven adaptation of system behavior with an evaluative objective function (ML). The approach of focusing on the problem description but not on the solution path has similarities to the paradigm of declarative programming, especially in GOFAI. However, only in some cases is finding a path to a solution essentially left to the computer, as in PROLOG, where the input database containing inference rules forms the basis for a rule-based (Depth-first) search for correct answers to queries. Furthermore, it should be taken into account that AI systems can be modular and combine functions that may have been generated using different approaches. Hereby, we achieve a relatively stable basis for a purposeful delimitation of the domain in our context. It also provides a clearer basis for explanations of the phenomenon domain from a computer science perspective.

3.3   Peculiarities of AI Systems

From **user-oriented perspective (U)**, the application of AI systems often seems familiar and simple at first. The verb "to google" refers to querying a well-known knowledge-based AI system and has already entered common usage. ML-generated software can besides other things make computers good at "hearing," "seeing," or natural language processing (e.g., translating text). Generative systems can use a few keywords or linguistic input to generate e.g. well-formulated essays, artistic-looking images, or program code for various questions in seconds. AI systems also can present adaptive or interestingly acting interaction partners in complex game-like environments. Chatbots based on GPT-4

may be almost indistinguishable from a human in short conversations [1]. Thus, they pass the "Turing test" in many cases [23]. Users without basic computer science education, even young children [22], can easily take on the role of developers ("trainers") with ML and create their own solutions to problems, such as gesture controls or intelligent software agents. However, in contrast to this often intuitive usability even for problem solutions, the interpretation and use of the outputs of these software systems require special skills, if one does not want to be exposed to undesired effects or cause them with one's products. ML systems produce "only" approximate solutions based on the presented examples, which partly require stochastic methods of interpretation ("confusion matrix"). Although in GOFAI the behavior of the system can be explained by studying the programmed logic [10], intuitive software production in the form presented above is not readily possible. In particular, ML systems have hidden limitations, including in [15] a number of "hard" problems: "one-shot learning", i.e., the ability to learn correct classification skills with only one or a few examples of a given class of objects, cross-domain generalization ability, causal inference, concrete meaning, "grounding", or the complexity of time scales and memory, and metacognition. Separately taught knowledge of these qualities is of paramount importance to all teachers because of the enormous impact they have on our personal lives [16].

In **social-cultural perspective (S)**, teachers (as well as students) need to be empowered to analyze the impact, opportunities, and challenges of AI. In addition, they need to know how to address potential problems in the use of AI to ensure responsible use (see Domain S [16]). To this end, it is also critical to clearly characterize the role of humans. There are some (ethical) grievances inherent in the technical nature of these systems [15]: for example, AI may not produce the intended performance or defensible reliability, produces biased or toxic results, violates privacy (or copyrights), produces false information about the world, exhibits a lack of explainability, contains consequences of lack of diversity in the people who research and develop AI in industry and academia, e.g., gender or race. AI systems are spawning a large number of new, but sometimes ethically dubious, tools and applications in the digitally networked world, including educational ones. The UNICEF AI definition [24], referred to among others in the update of the EU digital literacy framework "DigComp 2.2" [4] designates that the design and behavior of AI systems are also always subject to goals that have been determined by human system designers. This a fact that can be concealed by ostensible autonomy, objectivity, or by anthropomorphism, eloquence, or the like, whereby people without appropriate reflective competencies, especially children, can easily be subject to fatal deceptions here. AI systems produce a large number of new tools and possible applications, including in the field of education, which can be ethically questionable.

From a **technological perspective (T)**, AI systems do not add any new functions to the basic processes of digital transformation "capturing and storing" (digitization), "processing" (automation) as well as "transmitting and disseminating" (networking) information (cf. [7]), however, in the case of AI software,

the essential part of the information processing takes place through a process that has not been designed manually, but is produced by "facts" or examples and the objectives (cf. Sect. 3.2). Although the function of both types of AI is crucially based on the input data used, in practice ML systems stand out from GOFAI systems, in this respect, where the general inference algorithms search for the specified solutions in a space generated by the facts and rules, that can then be exactly traced [10]. Although the scope of application of ML systems has proven to be surprisingly extensive, on the other hand, the functionality generated in the training process by iterative approximation can only satisfy statistical quality criteria, which also clearly distinguishes the products of ML methods from those of "manual" software development, where humans use structured decomposition and analytical insight to pursue, among other things, the quality criterion of correctness, which is verified and, if possible, validated by various methods. In principle, most of the presented peculiarities of AI systems result from the way their information processing process was produced. Therefore, appropriate informatics education is also of central importance in establishing AI competencies.

4   AI-PACK - AI Competencies for Teachers

In this section, we will briefly describe each field and give some illustrative examples from each of the three perspectives of the "Dagstuhl Triangle".
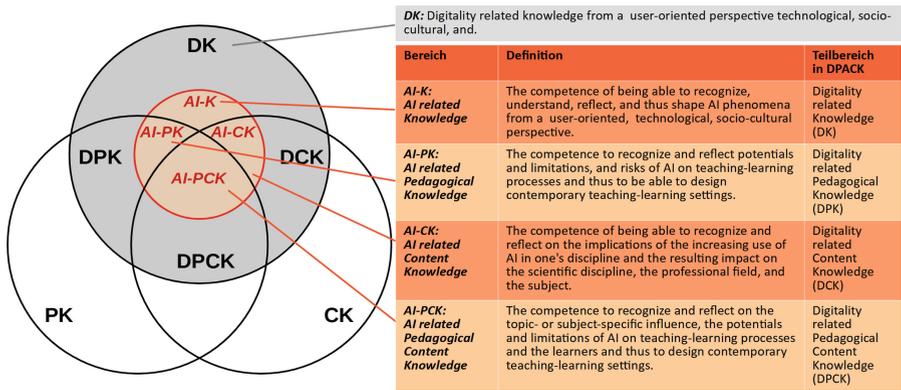
Fig. 1. AI-K with AI-PK, AI-CK, and AI-PCK within DPACK model.(The A was inserted at TPACK for stylistic reasons. We retain this for the name of the model).

4.1   AI-K: AI Related Knowledge

AI-K refers to the competence of being able to recognize, understand, reflect, and thus shape AI phenomena from a technological, socio-cultural, and user-oriented perspective.

If the focus is only on digital competencies, i.e. pedagogical and subject content competencies and their interactions do not play a special role, it is sufficient here to apply the Dagstuhl model. An application of the Dagstuhl model with respect to general, non-pedagogical, or content-related AI competencies in the context of CS K-12 education is available, with [16]. Accordingly, competencies would be to be able, for example, to critically question suggestions and prices (e.g., in online stores) as results of conscious and unconscious use (A-"How do I use this?"), to discuss reliability, e.g., in the context of self-driving cars (S-"What are the effects?"), or to select an appropriate ML procedure, e.g., to automatically recognize images containing certain artifacts (T-"How does this work?"). However, to perform their jobs, teachers still need some additional or more specific knowledge beyond what students are expected to acquire. Especially for Non-Computer Science contexts, the question of specific applications usually arises first. Linked to this are then questions of how the technology works and what the societal implications are.

4.2   AI-PK: AI Related Pedagogical Knowledge

AI-PK refers to the competence to recognize and reflect potentials and limitations, and risks of AI on teaching-learning processes and thus to be able to design contemporary teaching-learning settings.

The area comprises the general, non-subject-specific part of the AI competencies, which is necessary to be able to plan and implement lessons that are effective for learning. These competencies are located in the subarea DPK (cf. 1). "How can I teach (in general)'with','about', and'in spite of' the phenomena of artificial intelligence?" [8] This includes answers to questions such as "Where are my students currently in relation to digital media?", "How is digitalization currently changing society in general?", "How has the socialization of students changed, what opportunities, but also what problems and risks need to be considered with regard to teaching?" At AI-PACK, this is now applied to AI with its specifics. What are the implications of applications of the "AI world" from the students' environment like "TikTok", "Photomath", "DeepL", "Teachable Machine", "ChatGPT" or "Midjourney" etc. etc. for teaching in general? For teachers, a number of AI applications are also discussed to help with lesson planning, delivery, and reflection in general. Learning analytics is about "measuring, collecting, analyzing, and evaluating data about learners and their context with the goal of understanding and optimizing learning and the learning environment" (G. Siemens); in this context, great expectations are sometimes placed on corresponding AI applications.

A-"How do I use this?": This area is about the use of applications that allow e.g. the planning of effective lessons with a generative system, like ChatGPT, or the consideration of teaching forms that include adaptive self-learning systems that allow e.g. individualized and differentiated individual or group work or that analyze learning levels and progress as well as the use of reflection apps that support to reflect and process experiences.

T-"How does it work?": Following on from this, teachers should be able to roughly illustrate, for example, how ChatGPT was trained and produces its outputs, how apps with adaptive reward mechanisms work and respond to learners' attention and motivation, or how texts are classified or learning profiles are assessed.

S-"What are the effects?": Critically interpret and evaluate the outputs, outcomes, alerts, or notifications of AI tools and learning environments. What are the potentials, limitations, and risks for the learning group of such teaching-learning processes with adaptive self-learning systems that include personalized AI tutors or adaptive reward mechanisms? What is the impact of AI feedback on the learning group? What are the risks beyond the intended goals, e.g., with regard to privacy rights, such as data protection and copyright, or unfairness?

4.3   AI-CK: AI Related Content Knowledge

AI-CK refers to the competency of being able to recognize and reflect on the implications of the increasing use of AI in one's discipline and the resulting impact on the scientific discipline, the professional field, and the subject.

The area comprises the part of digital literacy (cf. 1) that is necessary to be able to teach a subject or a topic confidently: In what ways and with what procedures does AI come into play in subject-specific science, or how are its methods affected?", How are the corresponding professional fields changing because of AI systems?, "How is my subject being changed as a result?", "Does content disappear or is new content added?" cf. [8]. The starting point is corresponding subject-specific applications, e.g., the use of AI in reception or writing processes (language subjects), in translations (foreign languages), for the classification and explanation of artifacts (history), or in the identification of plants and animals based on photographs (biology). Corresponding competencies related to the aforementioned applications from the field would be,

... A-"How do I use it?": knowing and applying corresponding relevant professional AI tools.

... T-"How does it work?": to be able to describe how solutions are created technically, e.g. the classification of a plant image and what the differences are compared to traditional "manual" methods.

... S-"What are the effects?": to be able to evaluate AI solutions in a subject-specific way and to represent how such AI applications change the tasks and professional fields of experts in the subject. For example, what changes occur in history when applications are available that classify and explain artifacts such as images or writings? How reliable are the corresponding outputs and to what extent might the system be biased?

4.4   AI-PCK: AI Related Pedagogical Content Knowledge

AI-PCK refers to the competence to recognize and reflect on the topic- or subject-specific influence, the potentials and limitations of AI on teaching-

learning processes and the learners and thus to design contemporary teaching-learning settings.

In the DPACK model, the central area of "Digitality related Pedagogical and Content Knowledge" (DPCK) refers to knowing the most useful forms of presentation of relevant subject or thematic content, e.g., those of timeless and general importance, where aspects of teachability, instructiveness, and relevance (most meaningful analogies, illustrations, examples, explanations, and demonstrations) are also embodied [19] - understood under the conditions of digitality. It follows that teachers should be "digitally competent" in deciding what should be covered, how, and with what, i.e. without digital media or AI tools if necessary.

On the other hand, however, the area also refers to knowledge about the application possibilities of technology and pedagogical techniques with regard to targeted competency goals, as well as knowledge about how technology can help solve some of the problems that students face cf. [14]. In addition, with regard to the Dagstuhl perspectives, besides this application knowledge, there are also the skills to reflect on the digital means, in the case of AI-PCK the "AI software" technically and socially-culturally, to address and design them appropriately [8]. This means, for example, being able to,

... A-"How do I use this?": to generate subject-specific teaching materials or media involving AI (e.g., task variations, texts, images, videos, or simulations that include avatars, etc.) or to use appropriate AI-based tools to better convey subject content, e.g., to generate different text or translation variations and discuss them instructively in class (foreign language teaching) or to create instructive simulations or educational games using AI tools (e.g., in STEM subjects).

... T-"How does it work?": to describe how the applications mentioned work, i.e. to be able to explain how these systems have been trained and on the basis of which technical principles the outputs are generated.

... S-"What are the effects?": to assess the didactic value of e.g. self-evaluations using AI tools, such as chatbots, translators, tools that explain artifacts or "intelligent" math tools and to work through them appropriately with the students or to motivate subject content, if it is not to be dropped, even if perhaps tools exist that could take over these tasks, such as translators.

5   Discussion and Outlook

In our paper, we have presented a framework that enables the structured description and exploration of AI education requirements for contemporary professional teaching. With AI-PACK, we outline the AI-related domains of teacher professional knowledge AI-PK, AI-CK, and AI-PCK based on the DPACK model.

Our presentation narrows down the field of AI systems via their technical nature (design approach), which requires specific competencies. It is therefore based on an informatics perspective. A media pedagogical perspective may raise further issues, e.g. the problem that dealing with systems that feign human characteristics and abilities requires specific competencies, regardless of whether the

AI technology described in Sect. 3.2 was used, or that pretended "intelligence", as in the case of the famous "Mechanical Turk", is not generated by an informatics system but by covertly working humans (cf. [10]). The UNICEF AI definition [24], therefore includes, for example, systems that appear intelligent but are not AI systems from the technical perspective described here. CS education can be relieving if teachers understand how the outputs of AI systems are generated and how informatics problems are solved using AI methods. Many properties of the applications of the described area can be derived systematically and a more reflective handling of e.g. the outputs of such systems can be made possible. From a didactic point of view, understanding how AI processes subject-related data and thus builds up or applies its internal modeling could also provide new insights with regard to subject-related understanding, e.g. in comparison to corresponding manual processes.

Further research is needed with regard to the concretization, evaluation, and didactic design of the fields. On the one hand, the presented model shows strongly subject-related fields, whose evaluation and concretization require content-related and subject-didactic expertise (AI-CK and AI-PCK), but also interdisciplinary intersections, which are particularly well suited as subjects of interdisciplinary study offers, such as general methods of lesson preparation and implementation (AI-PK), as well as the obvious intersections in the T-areas ("How does it work?") and the related basics in computer science education. Therefore, in addition to the specific clarification of AI-PCK for CSE, we see the additional task for CSE to address the cross-curricular supplementary need for CS AI education in an appropriate way.

References

1. Biever, C.: ChatGPT broke the turing test - the race is on for new ways to assess AI. Nature **619**(7971), 686–689 (2023)
2. Celik, I.: Towards intelligent-TPACK: an empirical study on teachers' professional knowledge to ethically integrate artificial intelligence (AI)-based tools into education. Comput. Hum. Behav. **138**, 107468 (2023)
3. Commission, E., Centre, J.R., Redecker, C., Punie, Y.: European Framework for the Digital Competence of Educators: DigCompEdu. Publications Office (2017)
4. Commission, E., Centre, J.R., Vuorikari, R., Kluzer, S., Punie, Y.: DigComp 2.2, The Digital Competence Framework for Citizens: With New Examples of Knowledge, Skills and Attitudes. Publications Office of the European Union (2022)
5. Diethelm, I.: Digital education and informatics - you can't have one without the other. In: Proceedings of the 17th Workshop in Primary and Secondary Computing Education (WiPSCE 2022). Association for Computing Machinery, New York (2022)
6. Druga, S., Otero, N., Ko, A.J.: The landscape of teaching resources for AI education. In: Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education, vol. 1, pp. 96–102. ACM, Dublin (2022)
7. Döbeli Honegger, B.: Mehr als 0 und 1: Schule in einer digitalisierten Welt. hep, der Bildungsverlag, Bern, 2, durchgesehene auflage edn (2017)

8. Döbeli Honegger, B.: Das dpack-modell (2018). zugegriffen. https://mia.phsz.ch/DPACK/WebHome. Accessed 13 Jan 2023

9. Döbeli Honegger, B.: Covid-19 und die digitale transformation in der schweizer lehrerinnen- und lehrerbildung. Beiträge zur Lehrerinnen- und Lehrerbildung **39**, 411–422 (2021)

10. Holmes, W., Tuomi, I.: State of the art and practice in AI in education. Eur. J. Educ. **57**(4), 542–570 (2022)

11. Jung, Alexander: Machine Learning. MLFMA, Springer, Singapore (2022). https://doi.org/10.1007/978-981-16-8193-6

12. Karpathy, A.: Software 2.0. (2017). zugegriffen. https://karpathy.medium.com/software-2-0-a64152b37c35. Accessed 28 May 2023

13. Kim, S., et al.: Analyzing teacher competency with TPACK for K-12 AI education. KI - Künstliche Intelligenz **35**(2), 139–151 (2021)

14. Koehler, M., Mishra, P.: What is technological pedagogical content knowledge? In: Contemporary Issues in Technology and Teacher Education, vol. 9 (2009)

15. Manyika, J.: Getting AI right: introductory notes on AI & society. Daedalus **151**(2), 5–27 (2022)

16. Michaeli, T., Romeike, R., Seegerer, S.: What students can learn about artificial intelligence - recommendations for k12 computing education. In: Proceedings of World Conference on Computers in Education (WCCE 2022) (2022)

17. P21: Policy guidance on AI for children (2015). https://www.battelleforkids.org/networks/p21. Accessed 15 Apr 2023

18. Samoili, S., Lopez Cobo, M., Delipetrev, B., Martinez-Plumed, F., Gomez Gutierrez, E., De Prato, G.: AI watch defining artificial intelligence 2.0 (2021)

19. Shulman, L.S.: Those who understand: knowledge growth in teaching. Educ. Research. **15**(2), 4–14 (1986)

20. Sun, J., Ma, H., Zeng, Y., Han, D., Jin, Y.: Promoting the AI teaching competency of K-12 computer science teachers: a TPACK-based professional development approach. Educ. Inf. Technol. (2022)

21. Tedre, M., Denning, P., Toivonen, T.: CT 2.0. In: 21st Koli Calling International Conference on Computing Education Research, pp. 1–8. ACM, Joensuu (2021)

22. Tedre, M., et al.: Teaching machine learning in k-12 classroom: pedagogical and technological trajectories for artificial intelligence education. IEEE Access **9**, 110558–110572 (2021)

23. Turing, A.M.: I.-computing machinery and intelligence. Mind **59** (1950)

24. UNICEF: Policy Guidance on AI for Children (2021). https://www.unicef.org/globalinsight/media/2356/file/UNICEF-Global-Insight-policy-guidance-AI-children-2.0-2021.pdf. Accessed 13 Jan 2023

25. Yao, Y.: Deep integration of AI and TPACK: reconstruction of teachers' knowledge structure in the post-pandemic era. BCP Educ. Psychol. **3**, 150–154 (2021)

Implementing a Portable Learning Lab on Artificial Intelligence: It's AI in a Box!

Annabel Lindner[(✉)] , Marc Berges , Mathias Rösch, and Florian Franke

Computer Science Education and Nuremberg School Museum,
Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany
{annabel.lindner,marc.berges,mathias.roesch}@fau.de

Abstract. This paper describes the conception, design, and first evaluation attempts of a learning lab on artificial intelligence (AI). The learning lab, which consists of 25 learning activities, aims to teach the central concepts of AI and its applications in everyday life, industry, and research. To design the learning arrangements, major concepts of AI were selected based on the literature and made accessible to the students through playful experiments. In addition, research- and industry-related activities were created in cooperation with experts. In the research-led development process, prototypes of the learning activities were tested with students and improved based on their feedback. An evaluation concept was created and used to assess the final activities.

Keywords: Learning Lab · Artificial Intelligence · General Education

1 Introduction

With the release of OpenAI's ChatGPT[1] at the end of 2022, artificial intelligence (AI) has become a part of everyday life and social consciousness. AI systems that (seemingly) deliver impressive results are publicly available and easy to use for everyone. Besides the advantages of using such technologies, this also poses potential dangers: Now, even users with little or no knowledge of how the technology works can interact with AI systems. As a result, they might receive the products of AI systems less critically than necessary because the systems are by no means error-free (cf. [16,18]). The use of such systems is also attractive for students: essays, text translations, and even presentations can be completed almost magically with the help of AI systems (e.g., ChatGPT (See footnote 1) or DeepL[2]). Its generated results, which are presented very convincingly, are often adopted without reflection or further verification, and, in the worst case, false information and explanations are learned. To avoid such problems, all students should acquire basic knowledge about AI systems and how they work to enable them to deal appropriately and maturely with these computer

---

[1] https://chat.openai.com/chat.
[2] https://www.deepl.com/translator.

science phenomena [1] in their everyday lives. However, the speed with which innovative topics such as AI are integrated into school curricula, teacher training, and teaching materials often does not match the pace of technological progress. Consequently, in many cases, critical future competencies are only established in schools with a delay of several years. Nevertheless, in the meantime, the topic of AI can already be found in several curricula as it was included in recent reforms (e.g., grammar schools in Bavaria[3], AI4K12[4]). There are numerous initiatives and proposals to implement AI in secondary schools [12,14]. However, we are a long way from the goal of every student acquiring essential competencies in AI at school. So, we designed a portable learning lab on artificial intelligence to address this problem and close the development gap between technological progress and school. To be independent of school type, previous knowledge, age, or other specifics – we want to make AI concepts accessible for *every* student–, our learning lab uses 25 learning activities, both digital and unplugged, to teach fundamentals, applications, and research topics in AI. In this way, teachers are supported in introducing the topic of AI, which also brings new professional and educational challenges for the teachers themselves. The following sections describe the conception and structure of such a learning lab, illustrate example activities, and present the results of a pilot study on the lab's perception and assessment.

2   Research About Learning Laboratories

The term *learning lab* or *student lab* describes a broad range of out-of-school learning opportunities offered by different institutions, primarily in the STEM field. According to Haupt [3], these are permanent, out-of-school, or extra-curricular learning establishments that use special equipment for STEM subjects. The learning lab provides access to innovative and exciting topics beyond the curriculum and links them to students' personal experiences. In doing so, students are challenged to explore and act independently; the level of task difficulty is adapted to the target group. Classic learning labs are characterized by references to the curriculum and are visited together on a field trip. In contrast, portable learning labs visit schools with their equipment and organize regular activities under professional supervision ([3], https://www.schuelerlabor-atlas.de/kategorien).

According to Priemer et al. [15], the goals of learning labs include communicating the social significance of scientific content, reducing fears, and attracting students to the STEM field (cf. also [3]). In addition, lab goals can also be teacher-related by integrating them into teacher training and further education and passing on suggestions for lesson planning by using them. Furthermore, learning labs are frequently used as "laboratory schools" for educational research or places of science communication for companies and research institutions.

---

[3] https://www.lehrplanplus.bayern.de/fachlehrplan/gymnasium/11/informatik/ntg.
[4] https://ai4k12.org.

The influence and effectiveness of learning labs have been investigated in numerous studies, especially in the natural sciences, focusing on motivational factors and interest. A heterogeneous picture of the effectiveness of learning labs emerges. Concerning interest components, Priemer et al. [15] subsume in a meta-study that learning labs can increase students' interest initially but that these effects are short-lived. It is emphasized that a detached learning lab cannot achieve sustainable effects. Nevertheless, intensive preparation and follow-up of the visit must occur in the classroom context, in which Glowinski et al. [2], among others, observe a more durable interest. Itzek et al. [4] also emphasize that although visiting a learning lab contributes to a higher practical competence of the students compared to regular school lessons, it does not lead to a theoretical understanding of the contents and methods. Instead, it is necessary to integrate the learning lab visit into further teaching. This is also connected to the open and unstructured nature of the work in the learning lab compared to formal school lessons, in which the students achieve the best learning performance. Leiss [7] investigates the influence of learning labs on students' ideas in the field of physics and finds that learning labs can help to give students an up-to-date picture of research in the natural sciences and influence their ideas in this field.

3   Developing a Learning Lab About Artificial Intelligence

Existing computer science learning labs offer different activities on AI. For example, the Infosphere Laboratory at RWTH Aachen University includes a module on reinforcement learning for upper secondary school. The AI teaching-learning lab projects addressed by Lensing [8] are laboratories used in the context of higher education in engineering courses to link theory and practice. They have a very high degree of immersion as they aim to train subject experts. The laboratory presented in the following is dedicated to the subject area of AI, enables integration into lessons, and is set up directly at schools. Due to these characteristics, the laboratory is not an out-of-school establishment, as is typical for classic learning laboratories. Instead, it also aims to integrate innovative methods into learning at school and qualifies as a portable lab.

The designed learning lab pursues different objectives: Since previous research results (not only among students) indicate a very low level of knowledge about AI (e.g., [10]), basic AI-related competencies are addressed. The project wants to impart knowledge about AI, arouse interest and fascination for the technology and its future potential, and, at the same time, convey the urgent need for action in the field of AI in educational, political, and ethical terms. The students gain competencies for interacting with AI, but also for their future careers. The lab focuses on fundamental questions about AI such as: *Where do AI systems already affect everyday life? What can AI systems do in practical and theoretical terms? What should we be afraid of?* In addition to central concepts of AI (e.g., narrow and general AI, supervised learning, etc.), research areas, industrial and everyday applications, and social issues are equally included. For this purpose, experts from the respective disciplines participate in the conception of the learning lab, permitting different perspectives on the topic.

The learning lab also enables a holistic approach to the innovative topic of AI in the school context, independent of its curricular integration. The activities are designed to meet the requirements of different types of schools and different grades without being confined to a specific curriculum. This is realized through various accompanying materials and the possibility of differentiation within the scope of the activities. The learning arrangements of the lab, which always contain an information text in addition to the actual task and do not require any guided instruction, enable the students to explore the phenomena, questions, and applications of AI on their own and experimentally and playfully. During this experimental work, the students implicitly deal with the underlying concepts made concrete through the enclosed information texts.

The learning lab consists of 25 learning arrangements, each of which is integrated into an easily transportable wooden box of 60×60 cm. These can be attached to school desks so that an average classroom is sufficient. The boxes include unplugged tasks, games, and digital and technical elements. The materials developed for the learning lab, as well as the conceptual drawings of the boxes, are made available free of charge (https://www.kiki-labor.fau.de/) to enable teachers to replicate the boxes as well as to use the materials in class and to link lessons to (precedent) lab visits.

Besides the differentiation possibilities in content, the lab also shows variability in use. First, the boxes are placed randomly in the classroom and have no fixed order. Second, the participants can choose in which order they want to visit the boxes. However, teachers can provide a specific task for the visit, select which boxes are used, or set a predefined course through the lab. This is supported by the information texts and additional materials in each box, which provide information about thematically related boxes. In general, tasks are helpful to both motivate the reading of the accompanying texts and consolidate the concepts acquired in the lab or to prepare for the following lessons. Finally, the number of students interacting with a specific box can be varied (e.g., based on the students' age or task). Most boxes can be used by a single student or in groups of two or more students. However, some boxes are explicitly constructed as "multi-player" boxes and require at least two students.

Following the lab's aims, its limitation to 25 boxes, and a division into basic (Box 1–11) and application and research-related activities (Box 12–25), it is necessary to limit the included concepts to central, general educational aspects, which secondary school students can acquire self-directedly. To determine suitable topics and concepts, suggestions were systematically drawn from the literature [9, 11, 13, 17]. Table 1 outlines the central aspects taken as a basis for the subsequent development of prototypes for student-activating tasks. The concepts were coded at a rather general level (similar to the presentation in the respective papers) to correspond to the level of concepts aimed at in the laboratory activities. For the design of the activities and the corresponding texts, descriptions of competencies in the literature were attributed to the (knowledge) concepts they are based on. The current box-concept assignment is incomplete, as not all boxes have been fully planned and built yet.

Interested students tested these tasks in an informal setting (workshop, out-of-school, or in class): The students were observed by researchers while carrying

Table 1. Collection of AI concepts and competencies underlying the lab

Concept/Competency (subcodes indented)	Assigned Boxes
Recognizing AI systems	B1
Characteristics of AI systems	B1
AI vs. "normal" CS problems	B7
Defining AI	
Know & identify different applications of AI	B1, all application boxes
Understand the concept of intelligence	
History of AI	B8
Perceive the interdisciplinary nature of AI	all boxes
General & Narrow AI	B1
Know & understand different paradigms of AI	several basic boxes
Understand the programmability of AI	B2, B5
Knowledge Representation in AI systems	B13, B3
How are representations created?	B13, B3, B9
Reasoning and decision-making in AI systems	B13
Machine Learning (ML)	B8
Operating principles of ML	B2
Data Literacy	B6
AI systems learn from data	B2, B5, B3, B15
How does AI get from data to meaning/interpretation?	
Understand how AI systems use sensors to perceive	B9, B25
Understand that AI systems can act	B5
Limits, challenges & chances of AI	B10, B11, B4, B13, B12
Ability to assess results generated by AI systems	B10
Ethical Issues & AI's Impact on Society	B4, B23, B12, B14, B15, B18
Safety of AI systems	B11
Bias in AI systems	B4, B23, B16
Understand the concepts of explainable & transparent AI	B19
Human Role in AI	
Future development of AI	B20, B22
Differentiate correlation and causality with respect to AI	

out the activities, and difficulties, problems, questions, insights, and reactions to the tasks were noted. In addition, the students were asked how they liked the tasks, which aspects were unclear, which improvements and changes they would suggest, and what they had learned or thought they were supposed to experience. Based on these findings, the tasks were modified, refined, and retested.

The process is exemplary illustrated with the activity *Reality Taboo* (Box 9):

Underlying Concepts to be Taught: Reality is extremely complex and abstract. Processing and representing this complexity is difficult or impossible for machines or AI systems.

Activity Prototype: The students work in pairs; one student receives a picture and describes it to his/her partner. The partner makes a drawing based on the description. Certain words are "taboo" when describing, e.g., *horse, legs, head, ears, tail* for a horse picture.

Observations of the 1st Student Test:. Paraphrases are very easy to find. Moreover, complex, abstract terms can still be used. Therefore, it is not clear that machines do not rely on abstract concepts at all.

1st Revision: Instead of not being allowed to use certain terms, students are limited to the use of certain types of terms: Geometric shapes (examples are given), colors, types of lines, directions, and positions.

Observations of the 2nd Student Test: Examples of geometric figures confuse students if they do not know them (e.g., ellipse). In addition, the students should not choose the picture to be described themselves but always use the uppermost picture card. Otherwise, they only select easy motifs. A supervisor role can be introduced to control adherence to the term limitations if necessary. Concepts are now understood.

Final Revision: Minor conceptual adjustments to students' vocabulary and knowledge (geometric figures) and selection of final images for the box.

Following the prototype tests, a concept was developed to implement the respective activity in a wooden box. This involved working with a carpenter to create high-quality, stable, long-lasting materials. As part of the final design and manufacturing of the boxes, the materials were also professionally designed.

4   AI in a Box: Exemplary Description of Activities

The portable lab includes boxes that present applications of AI systems or current AI research and boxes that explain general functional/technical principles of AI. Furthermore, both unplugged and digital activities are used in the lab. Three boxes that represent these different types are presented in the following, the other activities are described on the website.

4.1   Application-Related Unplugged Box: *"Oracle-Cops"* (Box 23)

Inspired by the predictive policing AI software developed by the US company Geolitica[5] and used by US police forces, this box aims to illustrate how AI systems can be used to predict and prevent crimes. Students work on this activity in pairs or small groups. The box includes a large city map with labeled streets, activity cards, and blue and yellow tokens.

Fig. 1. "Oracle-Cops"

---

[5] https://geolitica.com.

Every activity card (cf. Appendix, Fig. 4 for examples) describes a property or incident related to a particular street on the map. The students' task is to discuss and decide whether the described incident or property can cause a crime in the respective neighborhood. If they assume this is the case, the place is marked with a blue token, otherwise with a yellow token. No token is placed if students expect no positive or negative effects of the described circumstance. After evaluating all events, the students decide, based on the ratio of blue and yellow tokens for each street, whether a police patrol should regularly visit it or not.

By performing this task, students take over the function of the AI system, which, based on past crime data and other socio-economic information about specific neighborhoods, also makes recommendations about preferable routes for police patrols. Their discussion of the "perfect" route and their evaluation of the incidents is essential to understanding that their subjective ideas and opinions are part of their decision and assessment. This also applies to AI systems when trained with data collected and generated by humans. After finishing the activity, the students read a text that explains how the Geolitica system works, how such algorithms can misjudge the conditions in specific neighborhoods, and which factors might contribute to the development of crime according to science.

The box represents an example application of supervised learning already used in everyday life and aims to convey the following ideas: Data stored in companies and government agencies can be used to train AI systems. With them, the systems learn to predict certain aspects, such as the probability of crime by concluding certain events from specific factors. This is not done objectively but is subject to human bias, as this bias is inherent in the underlying data and might also be caused by humans collecting and processing the data.

In a follow-up discussion in class, the activity can be used to illustrate AI applications in society and to discuss ethical aspects of AI: Which problems can arise from AI using data that is biased, i.e. includes stereotypes and prejudices? What might happen in disadvantaged areas when they become a police focus due to AI algorithms? Other application-oriented boxes in the lab that have already been completed present an approach to using AI systems for the control and early detection of epidemics, or take a look behind the scenes of the Spotify algorithm and illustrate how AI systems can be used in medicine.

4.2   AI Principle Unplugged Box: *"Wanted: AI"* (Box 1)

This box allows students to get to know phenomena from the field of AI and to delimit them from "normal" computer science applications. For this purpose, the box provides the students with wooden plates that depict everyday objects and applications (examples depicted in the Appendix, Fig. 5). In partner work, the students discuss whether the pictures show AI systems and arrange them accordingly in the box. The plates depict clear examples, such as a mixer,

Fig. 2. "Wanted: AI"

a printer, or a digital assistant, but also examples that encourage further discussions, such as spam filters or the control of power grids. On the one hand, the activity aims to show students' existing encounters with AI systems in their everyday lives and provide them with orientation points concerning the topic. On the other hand, misconceptions about using AI algorithms in certain products that students may have are unmasked. This is particularly important to help students develop an appropriate mental model of AI technology.

After arranging all example plates, the students can check their results with a UV flashlight as the recommended attribution is attached to the plate with a UV marker. In the following, the informational text describes why specific applications are AI systems or not and briefly explains how they work. Furthermore, it is highlighted that AI systems use particular kinds of algorithms (currently) created for one unique application and do not possess human skills like thinking. The activity examples can also be used for follow-up discussions in class: A definition of AI can be developed based on the applications, concrete AI methods can be explored based on the examples seen, and limits of AI applications in everyday life can be discussed or even tested with the objects and applications.

Besides Box 1 and the *Reality Taboo* presented in the preceding section, the lab includes several other unplugged boxes on AI principles and different AI paradigms. For example, students do a matching activity on the history of AI, create decision trees by using training and test data in a supervised learning setting, or train a robot to draw simple figures using reinforcement learning.

4.3   AI Principle Digital Box: *"Artist Unknown"* (Box 10)

Students working with this box participate in a quiz that presents them with art pieces (photographs, artworks, texts, music, videos) and asks them to assess whether the artist is a human or an AI system. The box consists of a touch screen on which the individual artworks are displayed and where students can vote. Music and sound files are available via headphones. After making their choice, the actual solution is revealed to the students, and they can take a closer look at the artwork again. Each quiz game consists of six art pieces, and several rounds can be played because new items are chosen randomly from an extensive collection of media in each round.

Fig. 3. "Artist Unknown"

This box aims to show students that AI algorithms can learn to generate creative artifacts. However, AI systems are not inherently creative but either use patterns that the algorithms have identified in human-created artworks to create new media or determine through trial and error what, for example, realistic photos look like. The corresponding informational text illustrates how AI systems can generate pictures using Generative Adversarial Networks (GAN) and shows, which details can help differentiate

between authentic and generated images, as it can often be challenging to identify digital media and art as AI-generated. Being able to assess and recognize the results generated by AI systems and to critically reflect on digital media represents an essential skill for the students in their daily lives as fake news and media are getting more common and are becoming harder and harder to identify. The achievements of AI systems, as well as the limits of their creations, can also be a topic of the lessons following the lab visit, especially since AI artifacts can be created easily and (mostly) free of charge on various websites[6]. Other boxes of the lab that use digital media permit creating deep fakes, exploring face recognition software, or observing how the output of AI systems varies based on the input data used for training. This includes the deliberate use of biased data sets that represent critical prejudices.

5   Evaluation

In the first evaluation, we wanted to know if the learning lab is perceived differently by students with a high and low affinity towards the subject of computer science (CS). Consequently, students' interest in CS was surveyed, as well as their computer science aptitude self-concept (pre-test survey). Following the lab visit, the students' self-concept concerning the learning lab itself was gathered.

A second goal of the evaluation was to find assessment methods that can be used permanently in the day-to-day use of the learning lab to enable a constant, easy-to-interpret evaluation of cognitive and non-cognitive facets of the students without disturbing the learning lab experience with questionnaires. For this reason, the assessment of students' interest was realized with a teacher questionnaire (questions on the students' CS skills and interests) on the one hand and a voting box survey for the students on the other hand. For this purpose, students were asked to throw a token (labeled with a number for each student to be able to link different surveys while securing anonymity) into one of five voting boxes corresponding to their CS interest (on a five-point Likert scale from "not at all interested" to "very interested"). By surveying the same aspect twice, the concordance between teacher and student perception was to be determined.

To answer the third question guiding the evaluation, namely to investigate whether fundamental concepts of AI are understood and remembered after being confronted with them in the lab, another non-questionnaire method was used: a digital quiz (Kahoot[7]) to test the students' conceptual knowledge after visiting the learning lab was piloted. The instrument included seven items in a single-choice form. A sample question looks like this:

AI-Systems...
     A: are always error-free due to the unambiguity of calculations
     B: know your personality and character based on your data
     C: collect data about you to make accurate forecasts
     D: are the more erroneous, the more data they get (multi-tasking error)

---

[6] e.g. DallE (https://labs.openai.com), Inferkit (https://app.inferkit.com/demo) or This Person Does Not Exist (https://thispersondoesnotexist.xyz/).

[7] https://kahoot.com/.

Concrete learning processes in the context of working on the boxes were not yet part of the evaluation and represent a starting point for future work. Aspects of students' behavior and interaction with the boxes were also examined during the workshops but cannot be presented here due to space limitations.

The self-concept evaluations were conducted with regular closed questionnaire surveys. Similar to the voting box, the data of the questionnaires were linked with numbers distributed to the students. Before visiting the lab, students answered four-point Likert-scaled items about their computer science aptitude self-concept based on Köller's scale [6]. After working with the lab activities, their self-concept concerning the learning lab was surveyed with four items (four-point Likert-scale, adapted from Kauper [5, p. 27]). Furthermore, we evaluated the students' motivation (12 items, Short Scale Intrinsic Motivation (KIM) [19]) to rule out a (negative) influence of motivation on the results of the self-concept studies. Since the survey took place in a German-speaking country, established instruments already available in German were used.

5.1   Results and Discussion

In 90-minute workshops, ten learning activities were used with students of a grammar school's 10th and 11th grades. 53 10th-grade students (13 female, 25 male, 2 diverse, 13 no information) and 18 11th-grade students (5 female, 11 male, 2 no information) participated in the survey. While cleaning the questionnaire data, eight incomplete data sets were excluded, resulting in the final evaluation of 63 data sets.

The teachers assessed the CS skills of all the participants as average. The students' interest in CS was rated as high in grade 11, corresponding to students' self-assessment in the voting box (M 3.66, SD 0.44). In the 10th grade, one class is rated as interested in CS, while the other is somewhat not. Here, students' assessment is contrary to the teachers': Class 1 (M 2.72, SD 0.74) rates its interest lower than Class 2 (M 3.3, SD 0.53), and both classes show some interest in CS. Therefore, the teacher's assessment of interest can not replace the survey of the students, since the perspectives deviate. This may be attributed to the respective teacher's concrete implementation of the CS lessons. Concerning the evaluation form, the voting box should be preferred over the teacher questionnaire as it allows individual instead of a global assessment of all students.

The self-concept (SC) items related to computer science aptitude [6] and the learning laboratory [5] were considered in the context of factor analysis; all items show medium to high loadings on the corresponding factor and can therefore be combined into one characteristic value each. On average, the students' CS aptitude self-concept is good (M 2.00, SD 0.67). The learning laboratory self-concept with a mean value of 3.08 and a standard deviation of 0.56 also shows a positive tendency. All motivational factors (interest/enjoyment, perceived competence, perceived choice, pressure/tension) are evaluated positively based on the mean values. The motivational factors correlate with the learning lab SC of the students at a weak to medium level ($r_{Interest} = 0.41$, $r_{Competence} = 0.62$, r_{Choice}

$= 0.25$, $r_{Pressure} = -0.18$). Thus, it can be seen that a high motivation of the students during the task is essential to achieve positive effects on the learning laboratory-related self-concept.

In the context of the learning lab, a particular focus is on whether this instruction can also reach students whose aptitude score in CS is relatively low. So, the results were divided into four groups: students with high or low computer science aptitude ($M < 2.5/ >= 2.5$) and high or low learning lab SC ($M > 2.5/ <= 2.5$). The results showed that seven of the nine students who rated their CS aptitude as low had a positive self-concept about the learning lab and got along well. Thus, the learning lab seems suitable for inspiring students who rate their basic computer skills as low. However, ten out of 54 students with a positive CS attitude evaluate their work experience with the learning lab negatively. The cause of this cannot be answered with the available data: This tendency is not dependent on gender, group, or CS interest. This group possibly evaluates the learning lab as not computer science-oriented enough due to its focus on unplugged or playful elements. However, this cannot be confirmed without further investigations.

Using the Kahoot quiz, the students show their knowledge without a test or survey character, and the short-term competition situation motivates them. A quiz of this kind can thus be integrated as a permanent component of the learning laboratory. However, testing conceptual knowledge presents a challenge regarding content: The piloted items currently do not have the required discriminatory power and therefore need to be revised regarding wording and distractors. In addition, the items have very different levels of difficulty.

In the context of these results, no meaningful correlations between self-concept and conceptual knowledge can be established. Thus, it is also impossible to conclusively state whether the intended concepts are understood in the learning laboratory. However, the overall high number of correctly answered questions is a positive indication. In addition, the informal observations during the school tests of the box design also indicate that working through the learning activities facilitates the understanding of central concepts of AI.

6   Conclusion

The pilot study shows indications that the learning lab can also reach students with a low aptitude self-concept and make CS topics accessible. The influence of the learning lab on the students' conceptual knowledge of AI cannot be assessed at the moment due to the low significance of the results of the knowledge items. Still, it must be investigated in more depth in further studies. In the context of additional surveys, an iterative improvement of the existing items by adapting the wording and revising the distractors is therefore intended, as well as the addition of further items for the other boxes, to finally clarify whether fundamental concepts of AI can be taught with the help of the learning lab.

A   Appendix: Sample Material from the Boxes

Fig. 4. Sample incident cards from "Oracle-Cops"

Translation from left to right:
(1) There are many old, rotten buildings in this neighborhood.
(2) There's a lot of unemployment on this street.
(3) This street has a playground, a mall, and a gym.
The letter refers to the street and is equally represented on the map.

Fig. 5. Sample picture cards from "Wanted: AI": autonomous vehicle, calculator, kitchen machine, face recognition. Images under CC-License, detailed resources on the website: https://www.kiki-labor.fau.de/.

References

1. Diethelm, I., Hubwieser, P., Klaus, R.: Students, teachers and phenomena: educational reconstruction for computer science education. In: Proceedings of the 12th Koli Calling International Conference on Computing Education Research, pp. 164–173. Association for Computing Machinery, New York (2012). https://doi.org/10.1145/2401796.2401823

2. Glowinski, I., Bayrhuber, H.: Student labs on a university campus as a type of out-of-school learning environment: assessing the potential to promote students' interest in science. Int. J. Environ. Sci. Educ. **6**, 371–392 (2011)

3. Haupt, O.J., et al.: Schülerlabor - Begriffsschärfung und Kategorisierung. Der mathematische und naturwissenschaftliche Unterricht **66**(6), 324–330 (2013)

4. Itzek-Greulich, H., Flunger, B., Vollmer, C., Nagengast, B., Rehm, M., Trautwein, U.: Effects of a science center outreach lab on school students' achievement - are student lab visits needed when they teach what students can learn at school? Learn. Instruct. **38**, 43–52 (2015). https://doi.org/10.1016/j.learninstruc.2015.03.003

5. Kauper, T., Retelsdorf, J., Bauer, J., Rösler, L.M., J Prenzel, M.: Skala: 4 Motivationale Orientierungen, 4.1 Selbstkonzepte, 4.1.1 SK Fach, p. 27 (2012)

6. Köller, O., Schnabel, K.U., Baumert, J.: Der Einfluss der Leistungsstärke von Schulen auf das fachspezifische Selbstkonzept der Begabung und das Interesse. Zeitschrift für Entwicklungspsychologie und Pädagogische Psychologie **32**(2), 70–80 (2000). https://doi.org/10.1026//0049-8637.32.2.70

7. Leiß, F.: Untersuchung von Schülervorstellungen über Tätigkeiten von Naturwissenschaftlern und deren Beeinflussung durch ein Schülerlabor. Dissertation, RWTH Aachen University, Aachen (2019). https://doi.org/10.18154/RWTH-2020-07442

8. Lensing, K.: Labore in der Hochschullehre. chap. Künstliche Intelligenz im Lehr-Lernlabor, pp. 263–282. wbv (2020). https://doi.org/10.3278/6004804w263

9. Lindner, A.: Designing a teacher PD programme for AI - first steps. In: The 16th Workshop in Primary and Secondary Computing Education (WiPSCE 2021). Association for Computing Machinery, New York (2021). https://doi.org/10.1145/3481312.3481350

10. Lindner, A., Berges, M.: Can you explain AI to me? Teachers' pre-concepts about Artificial Intelligence. In: 2020 IEEE Frontiers in Education Conference (FIE), pp. 1–9 (2020). https://doi.org/10.1109/FIE44824.2020.9274136

11. Long, D., Magerko, B.: What is AI literacy? Competencies and design considerations. In: Proceedings of the 2020 ACM Conference on Human Factors in Computing Systems (CHI 2020), pp. 1–16. Association for Computing Machinery, New York (2020). https://doi.org/10.1145/3313831.3376727

12. Miao, F., Services, K.: K-12 AI curricula-mapping of government-endorsed AI curriculum (2022)

13. Michaeli, T., Romeike, R., Seegerer, S.: What students can learn about artificial intelligence - recommendations for K-12 computing education. In: Proceedings of IFIP WCCE2022: World Conference on Computers in Education. Hiroshima (2022)

14. Micheuz, P.: Approaches to Artificial Intelligence as a subject in school education. In: Brinda, T., Passey, D., Keane, T. (eds.) OCCE 2020. IAICT, vol. 595, pp. 3–13. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59847-1_1

15. Priemer, B., Pawek, C.: Out-of-school stem learning in Germany: can we catch and hold students interest? In: 2014 NARST Annual International Conference, Pittsburgh (2014)

16. Roselli, D., Matthews, J., Talagala, N.: Managing bias in AI. In: Liu, L. (ed.) Companion Proceedings of the 2019 World Wide Web Conference, pp. 539–544. ACM Digital Library, Association for Computing Machinery, New York (2019). https://doi.org/10.1145/3308560.3317590
17. Touretzky, D., Gardner-McCune, C., Martin, F., Seehorn, D.: Envisioning AI for K-12: what should every child know about AI? In: The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI 2019). Association for the Advancement of Artificial Intelligence (2019). https://doi.org/10.1609/aaai.v33i01.33019795
18. Weitz, K., Schiller, D., Schlagowski, R., Huber, T., André, E.: "Let me explain!": exploring the potential of virtual agents in explainable AI interaction design. J. Multimodal User Interfaces (2020). https://doi.org/10.1007/s12193-020-00332-0
19. Wilde, M., Bätz, K., Kovaleva, A., Urhahne, D.: Überprüfung einer Kurzskala intrinsischer Motivation (KIM). Testing a short scale of intrinsic motivation. Zeitschrift für Didaktik der Naturwissenschaften **15** (2009)

Investigating the Role of ChatGPT in Supporting Text-Based Programming Education for Students and Teachers

Markus Wieser[1(✉)], Klaus Schöffmann[2], Daniela Stefanics[2], Andreas Bollin[1], and Stefan Pasterk[1]

[1] Institute of Informatics Didactics, University of Klagenfurt, Klagenfurt am Wörthersee, Austria
{markus.wieser,andreas.bollin,stefan.pasterk}@aau.at
[2] Institute of Information Technology, University of Klagenfurt, Klagenfurt am Wörthersee, Austria
{klaus.schoffmann,daniela.stefanics}@aau.at

Abstract. Teaching text-based programming poses significant challenges in both school and university contexts. This study explores the potential of ChatGPT as a sustainable didactic tool to support students, freshmen, and teachers. By focusing on a beginner's course with examples also relevant to vocational schools, we investigated three research questions. First, the extent to which ChatGPT assists students in solving and understanding initial examples; secondly, the feasibility of teachers utilizing the chatbot for grading student solutions; and finally, the additional support ChatGPT provides in terms of teaching. Our findings demonstrate that ChatGPT offers valuable guidance for teachers in terms of assessment and grading and aids students in understanding and optimizing their solutions.

Keywords: AI and Machine Learning · Pedagogy/Teaching Approach · Programming Education · Higher-Secondary and Vocational Schools

1 Introduction

Teaching text-based programming is a challenge that requires a lot of practice, reflection, and thus time. It involves technical knowledge and understanding of the fundamental concepts needed to solve programming problems. In this context, there is currently hype around ChatGPT (and similar tools based on large-language models) that teachers, students, and learners use to write texts, formulate papers, and solve programming tasks [7].

ChatGPT was initially developed as a text generation tool and is known for its ability to produce human-like text. This development has led to its use for

solving programming tasks, as well. It may be seen as a tool for cheating, but with its text and program generation skills, it provides many teaching opportunities [1]. Of course, there are reasonable doubts about the quality of the results provided by ChatGPT, and some experts refer to such tools as stochastic parrots [2], as they merely assemble prefabricated text modules and do not perform actual problem-solving. This issue may be a problem in the classroom. When using ChatGPT as a tool during a lesson, students need help recognizing incorrectly generated answers or solutions. Lack of knowledge can lead to the internalization of wrong concepts, preventing the development of new competencies [9].

With this in mind, this paper is dedicated to the quality of support ChatGPT offers students and teachers when using it. It is about recognizing the limitations of this tool and understanding how it can best be used to support learning and understanding of programming. In particular, to better understand these issues, we selected examples and solutions from a beginners programming course at University of Klagenfurt that is also part of the computer science curriculum at vocational schools in Austria, and we investigate the following research questions:

– RQ-1: How much help does ChatGPT provide pupils to solve practical exercises?
– RQ-2: Can teachers use ChatGPT for grading solutions of the pupils?
– RQ-3: What other programming teaching assistance does ChatGPT offer?

The paper is structured as follows. Section 2 looks closely at publications on teaching programming and using ChatGPT in the school and university context. Section 3 provides an overview of the different possibilities that ChatGPT can be used in the classroom. Section 4 presents our study to evaluate the quality of the tool's responses. Section 5 presents our didactic recommendations, and Sect. 6 concludes the contribution with a summary and a short outlook.

2 Related Work

If we want to test the usefulness of a new tool in a school (or university) context, we must consider how the learning process takes place. An abundance of resources and activities can be found to master text-based programming.

Scientific work on teaching programming focuses on understanding and supporting the cognitive development of novice programmers. Research by Lister et al. explores the application of Bloom's and SOLO taxonomy to differentiate programming tasks and assess programming skill levels [13,14]. Investigations of the hierarchical development of programming skills, including reading, tracing, and writing program code, provide valuable educational insights [12]. Neo-Piagetian developmental stages offer a framework for understanding novice programmers' cognitive abilities and transitions, emphasizing the importance of aligning programming tasks with students' cognitive development [20].

In the realm of programming education, a lot of strategies have been explored. These include explicit teaching of problem-solving strategies [16], understanding novice programmers' mental models [3,15], evaluation of problem-solving

techniques in Java [6], and challenges faced by low-skilled students in code comprehension [19]. Studies also highlight the significance of detecting programming skills early [10], the analysis of sketch types in code reading tasks [4], learning trajectories for programming concepts [17], and the utilization of Parson's Problems as a skill acquisition tool [5]. Together, these diverse studies offer a comprehensive view of effective strategies and techniques for teaching programming.

Large language models, like GPT-3.5 or Open-AI Codex, are increasingly being utilized in programming education, and they are fascinating and terrifying simultaneously [8]. They offer an interactive learning environment and can assist in various tasks. They are used to improve error messages [11] and they are used to create programming exercises as well as explanations [18]. However, at the time of writing this paper, there are no articles looking closer at the quality of the results and their influence on teaching programming. As a contribution to existing knowledge, our work, therefore, takes a closer look at typical application areas and also illuminates them from a didactic perspective.

3 Quality Considerations

3.1 Setting and Methodology

We selected the *Introduction to Java Programming* lab course at University of Klagenfurt (winter term 2023), which is part of the *Informatics* curriculum (besides a few others) and had about 120 students enrolled. The lab course required students to solve 103 Java exercises (distributed over 10 assignments) that had to be submitted via the CodeRunner plugin of Moodle, which checks every exercise with a few test cases (and often some pre-defined code, where the student submission is embedded). The lab exam is also conducted via Moodle/CodeRunner, and consists of programming exercises only. As such, the exercises and tasks are comparable to tasks in the last two classes of vocational schools in Austria, and the results should also be applicable in the school context.

Our experiment, which is performed with ChatGPT 3.5, has two parts, aside from just letting ChatGPT solve the programming tasks. In the first part, we want to check whether ChatGPT could be used as a personal tutor; in the second part, we check its grading and assessment capabilities.

Solving Programming Tasks: For the evaluation of RQ-1, we copy the description of every exercise from Moodle to the ChatGPT prompt, check the response, and copy it back to the CodeRunner input window, where it is submitted and automatically checked with our test cases. In case the first provided solution of ChatGPT does not meet the requirements, we ask to regenerate the solution and provide more necessary details (or small change requests).

Grading and Assessment: For the evaluation of RQ-2, we ask ChatGPT first to assess students' solutions to eight defined tasks. We use the solutions from an exam at the end of the course, in which a total of 84 students participated (for eight different tasks to solve). From all of the solutions, we end up with 672 code snippets to analyze. We give ChatGPT the respective task definition and solutions and ask it to score each solution up to a given maximum of points. We then analyze the results and compare them to the points the teachers have given, using the Spearman Correlation. We use this method because the data is not normally distributed. Furthermore, we select a random group for each of the eight tasks, for which we ask ChatGPT to assess the tasks two more times in different chats, and we finally compare the consistency of the assessments.

We also take a sample of five students' solutions to the four tasks with the highest possible points. We select those with the highest difference between teacher and ChatGPT. When there are more than five, we randomly select some. We then give the teachers three statements to be evaluated on a five-point Likert scale. In addition, there is the possibility to submit explanatory notes to each question.

Another aspect we are interested in is the code understanding of ChatGPT compared to a teacher. Therefore, we define four types of error that are typical for beginners of the Java programming language: (1) syntax errors, (2) runtime errors, (3) logical errors, and (4) semantic errors. We then compare the error analysis of ChatGPT with the teacher's analysis.

Individualized Instruction: To get tasks adapted to the learning needs of students or whole groups, we present ChatGPT with the student's submissions and ask it, based on the mistakes, to create tasks that could be used by the students to specifically practice these topics. We use two runs: in the first run we just present ChatGPT the task and the student's solution and ask how a possible task could look like, so that the students could improve their skills. In the second run, we also give suggestions, such as: "It seems that the student has problems with indexing." We give five exemplary tasks to ChatGPT. Each task is submitted once as a single submission and another time as a group submission.

4   Findings and Recommendations

In this section, we will present and discuss the results for the areas *Solving Ability*, *Grading and Assessment* and *Individualised Instruction*. We briefly introduce each topic and close the respective section with didactical recommendations.

Some papers are currently discussing the possibilities of ChatGPT in education. As Zhai [21] points out in the paper focusing on academic writing, AI is certainly capable of supporting students and teachers in a wide range of tasks. Some of these areas are also relevant to the teaching and learning of programming. In our experiment, we found that ChatGPT, aside from being a simple solving tool, can be an asset to programming education in other areas.

4.1   Solving Ability

Results: From a total of 103 Java programming exercises, 100 (97.09%) could be correctly solved by ChatGPT. For the vast majority of exercises (59.22%), already the first provided Java code could be directly submitted via CodeRunner and all test cases succeeded. Another 28.16% of exercises could be quickly solved by a simple *Regenerate response* request, so that the second proposed solution was correct. For another 9.71% of exercises, some further minor changes were required, which were successfully performed by ChatGPT after we provided more details. Examples of such changes are extreme situations that are checked by CodeRunner test cases (empty arrays), unnecessary Getter and Setter methods (that needed to be removed), wrong types of loops (e.g., for-each loops instead of normal for loops), and wrong naming of variables or methods. Figure 5 (see Appendix) shows such a code example where a CodeRunner test case failed for the parameters (null, 0). A request was then made to ChatGPT that the code should work for these parameters as well and it adapted its solution to what is shown in Fig. 6 (see Appendix). With this solution all tests finally succeeded. Only 2.91% of all exercises could not be solved by ChatGPT.

Discussion: It should come as no surprise that ChatGPT is able to solve almost all the tasks of a beginner's course in programming. If the solution does not meet the requirements, adding some minor changes will solve the problem. Even at this stage, ChatGPT could be considered as "Solution for all problems", at least as far as the beginners' courses in programming are concerned. So, from now on it is no longer possible to measure the students' learning progress on the basis of their submissions to assignments (or to determine what their abilities and weaknesses are), since the solutions could have been generated by generative AI tools. But do harder tasks solve the problem? In this case, the tasks would have to be so complex that Chat GPT cannot properly solve them. However, such tasks are definitely not suitable for a beginner's course in programming.

Didactical Recommendations: The very existence of ChatGPT now creates a major problem for all teachers in programming: How can I correctly assess and evaluate the skills and abilities of my students? What part of their tasks is made by themselves and where did they use tools like ChatGPT? Unfortunately, we have to assume that ChatGPT will solve all tasks of this kind in the future, and if we continue to set tasks as we are now, there will be no learning progress for students. So there have to be other methods of measuring the students' abilities in programming. First, there is the possibility of writing tests and exams. In such a situation, it is much harder to use unauthorized tools and so the skills and weaknesses are shown. On the other hand, there are students with exam anxiety. For those such a mode would be horrifying. Another possibility is to test the students' abilities in class. In our University the students get weekly task sheets and they then hand in the solved tasks week by week. In classes, students are randomly selected to present an example they have solved. It would not be a big deal to let them solve very similar tasks instead of the task already

solved. There are several advantages. First, because they do not know what task is given to them in the exercise, the students are not able to just present a ChatGPT solution and therefore have to understand the concepts and learn programming by themselves. For students who are always learning along, these changes do not pose much of a challenge, as long as the new task actually differs only slightly from the original task.

4.2   Grading and Assessment

The quality of a program depends on different criteria, such as functionality, efficiency or correctness. Assessing a program according to these criteria and grading the student accordingly can be very time-consuming and complex, depending on the number of students and the size of the program. Here, ChatGPT could be used to automatically grade assignments based on criteria defined by the teachers.

Results: When looking at the results of this analysis, we were pretty much surprised. In general, the grading of ChatGPT was pretty useful. The awarding of points was coherent and comprehensible at first sight. It also explained the awarding scheme (see Fig. 7 in Appendix).

We computed the Spearman correlation for the teacher's points in comparison to the points awarded by ChatGPT, separate for all tasks. We used Spearman because the data is not normally distributed. The results received are shown in Fig. 1. As visible in the figure, there is a maximum correlation of 0.951, which is pretty strong, and a minimum correlation of 0.678, which also indicates a medium to strong positive relationship between the ranks.

When we then compared the assessments of ChatCPT to each other, we found that ChatGPT's assessments are not constant and accordingly not always the same. In Fig. 2 the Spearman correlation between the three different attempts (A1,A2,A3) and the respective teacher (T) is shown.

Task	Ex 1	Ex 2	Ex 3	Ex 4	Ex 5	Ex 6	Ex 7	Ex 8
r	0.844	0.727	0.840	0.678	0.839	0.934	0.951	0.905

Fig. 1. Spearman Correlation of the different tasks

1 - D	A1	A2	A3	2 - C	A1	A2	A3	3 - A	A1	A2	A3	4 - E	A1	A2	A3
T	0.822	0.771	0.833	T	0.861	0.861	0.881	T	0.936	0.843	0.991	T	0.529	0.684	0.526
A1		0.704	0.704	A1		1.000	0.978	A1		0.911	0.927	A1		0.688	0.773
A2			0.989	A2			0.978	A2			0.843	A2			0.930

5 - D	A1	A2	A3	6 - B	A1	A2	A3	7 - C	A1	A2	A3	8 - F	A1	A2	A3
T	0.932	0.937	0.758	T	1.000	0.974	0.974	T	0.861	0.978	1.000	T	0.778	0.710	0.869
A1		0.868	0.687	A1		0.974	0.974	A1		0.947	0.861	A1		0.767	0.743
A2			0.899	A2			1.000	A2			0.978	A2			0.579

Fig. 2. Spearman Correlation of different tasks including ChatGPT attempts

The correlation between T and A1 differs from the correlation in Fig. 1, because here we only used one exam group instead of the entire population. As you can see, different grading attempts return different results, although the input was the same. When going into detail, we found that ChatGPT gave the maximum amount of points and the minimum amount of points to the same submission in different grading rounds. So on one attempt, it gave 10 Points to the student, and on another attempt, it gave 0 Points.

When evaluating the selected student's solutions and ChatGPT's explanations, we found some pretty interesting things. We took the five solutions of each task with the highest difference between ChatGPT and the teacher's assessment and then asked the teachers to evaluate three statements by five-point Likert scale, where 1 corresponds to *does not apply* and 5 corresponds to *applies* for each solution.

1. ChatGPT's awarding of points is coherent and comprehensible
2. Based on ChatGPT's scoring, I would change my own assessment
3. The assessment only by ChatGPT would have led to a reasonable result for this solution

The mean of the answers can be found in Fig. 3. The number of the item corresponds to the number of the question.

Mean	Item 1	Item 2	Item 3
Ex 2	2.20	1.40	2.20
Ex 3	1.00	1.00	1.00
Ex 5	1.40	1.40	1.60
Ex 8	2.25	2.00	3.00
Total	1.68	1.42	1.89

Fig. 3. Evaluation of the Likert scale of all three items

As we can see, the teachers share the opinion that ChatGPT's awarding of points is neither coherent and comprehensible (1.68/5), nor do they want to change their own assessments (1.42/5), nor do they think the assessment only by chatGPT would have led to a reasonable result for this solution (1.89/5). When we take a closer look at the answers of the teachers, then we find, that ChatGPT ignored some mistakes. So the explanation of one of the teachers was: "ChatGPT overlooks some errors here: (1) the current element is never compared with the minimum, but with the neighbor, (2) array index and value are confused (in the variable min the index is stored instead of the value), (3) the initialization of min is syntactically wrong. The reason for the deduction of only 1 point (loop condition) is not comprehensible."

Regarding the error analysis capabilities of ChatGPT we computed statistics on the five error types for five programming exercises of a programming exam where 84 students participated. It is important to note that the teacher checked for syntax and runtime errors first and did not look further if one of these errors
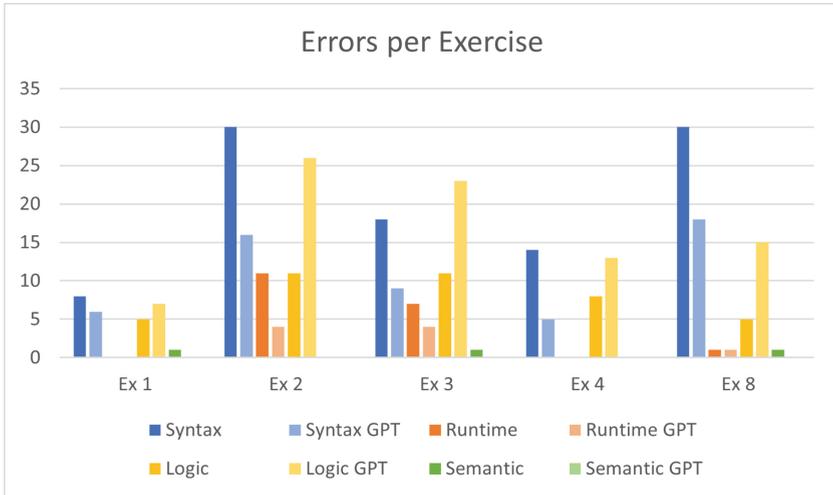
Fig. 4. Comparison of errors per exercise

was detected. ChatGPT on the other hand always checked for several types of errors and mentioned all of them in a list. However, for statistics, we only counted the first error type mentioned for the teacher and ChatGPT. The overall sum of errors is similar: 162 errors were detected by the teacher, while 147 were detected by ChatGPT. The total number of errors for each of the five exercises is also very similar (Teacher vs. ChatGPT): Ex 1: 14 vs. 13, Ex 2: 52 vs. 46, Ex 3: 37 vs. 36, Ex 4: 22 vs. 18, and Ex 8: 37 vs. 34. When taking a closer look at the different types, the statistics reveal that the teacher detected 100 syntax errors, 19 runtime errors, 40 logical errors, and 3 semantic errors, while ChatGPT found 54 syntax errors, 9 runtime errors, 84 logical errors, and 0 semantic errors (see Fig. 4). The difference in syntax and logical errors can be explained by the different prioritisation as described above: the teacher did not further assess the solution if it did not compile (e.g., wrong syntax, incomplete code, etc.) or crash (e.g., IndexOutOfBoundsException), while ChatGPT inspected the code in detail. As the sum of both error types is very close (140 vs. 138) and the teacher agreed after another check of several solutions that many solutions with syntax errors also include logical errors, we can conclude that the assessment of ChatGPT is very accurate in our qualitative evaluation. The same is also true for runtime errors. For example, in one exercise, the student used the following code for an inner for-loop: `for (int j=0;i<arr2d[i].length; j++)`. This led to a run-time error, and this is what the teacher counted. ChatGPT can also detect the run-time error but first mentions the logical error that variable i is used instead of variable j for the check, which is the type of error we counted for the statistics. Semantic errors were classified by the teacher when incorrect operators were used (e.g., the XOR operator instead of Math.pow()), or when students used any fixed hard-coded values instead of variables. These errors

were also found by ChatGPT but classified as logical errors. Overall we can summarize that the error analysis by ChatGPT is very accurate and complete, no single error could be identified that was not revealed by the chatbot as well.

Discussion: When taking a first look at the results of the first assessment attempt we found, that the grading of ChatGPT was strongly correlated to the assessment of the teachers. We had a maximum Spearman correlation of 0.951 and a minimum correlation of 0.678. Both are strong correlations, the higher one even *very strong* (see Fig. 2). So from a statistical point of view everything was fine. The problem is, grading and assessment is not statistical, it is individual. The teacher's job is to determine if a student has achieved the learning objectives or competencies and where the weaknesses and strengths are. And such a task can not be performed by a statistical model that assigns zero points to a solution in one prompt, and full points on the same submission in another prompt. This inconsistency in grading disqualifies ChatGPT as a standalone grading and assessment tool. Another problem here is, that ChatGPT overlooked some mistakes and, according to the teachers, gave non comprehensible assessments, on the other hand it was pretty accurate in finding errors and error types.

Didactical Recommendations: In summary it could be said, that ChatGPT should not be used as an unsupervised grading tool. In our opinion, the negative aspects, such as incorrect assessment or inconsistent scoring, outweigh the statistical correlation. But it could be used as an *alternative opinion*. If you have a submission additionally checked by ChatGPT, you will get a further error analysis and a score hint if needed. This could help to find errors that were overlooked by oneself. However, it should be noted that such a procedure could require additional time resources. The assessment and grading itself should definitely continue to be done by the teacher.

4.3   Individualized Instruction

Individualized Instruction offers several benefits to schools and learners alike. It enables students who may be above or below average in their learning abilities to proceed at their own pace, ensuring optimal learning outcomes. This personalized approach allows students to avoid repeating portions of a course that they have already mastered, leading to a more efficient use of time and resources. In this area, ChatGPT could take on the function of a personal tutor who, based on the available data, provides individual explanations and error analyses.

Results: When testing the tutoring abilities of ChatGPT, we found, that its general recommendations without a defined learning goal were pretty generic. In these cases, the suggestion of ChatGPT is mostly to solve the same example again, only it adds "Additional Guidelines". For example, if the student has simply specified the length in the form of hardcoding instead of array. length, the additional guidelines say: "avoid hardcoding". However, there were also suggestions that were not suitable despite the learning objective being pointed out.

For example, one suggestion from ChatGPT was that the student should just fix their mistakes. In other suggestions it wasn't really clear what the student was supposed to do. However, the majority of ChatGPT's suggestions after defining the learning objective was quite good. Most of the time ChatGPT let the student solve the same or similar example again, but gives detailed solution paths. We consider these examples suitable, because the student can work through the example again step-by-step and thus get an idea how one could approach such tasks. In the group comparisons, there is sometimes even a separate task for each student to practice, which addresses the individual errors of each individual and then another common task.

Discussion: As already pointed out in the paragraph above, the general recommendations of ChatGPT are generic. If students want to practice and therefore know their exercise potential, recommendations like doing the same task again, just fix the mistakes or avoid hardcoding are not useful and in fact demotivating. If a learning goal is submitted, the recommendations are more useful and therefore could be used in individual practicing, combined with step-by-step solutions and detailed solution paths. Here, ChatGPT could be used as an individual tutor, giving the students the opportunity of working through a different example or getting explanations on their own pace.

Didactical Recommendations: When trying to use ChatGPT as personal tutor there are some limitations. In order to get useful results, the request should be defined by the teacher. So the teacher should tell each student, which subject areas need further practice and how the request should be made. The students then should be able to practice their learning needs with the aid of ChatGPT.

5   Conclusion

Our research questions can now be answered as follows. Concerning question (RQ-1) "How much help does ChatGPT provide pupils to solve practical exercises?" we found that unsurprisingly ChatGPT is in fact capable of solving all the tasks given in a programming beginner's course. This problem could be solved by increasing the number of tests and exams or challenging the students in class using slightly modified tasks. Concerning question (RQ-2) "Can teachers use ChatGPT for grading solutions of the pupils?" we found that ChatGPT could be used as a grading and assessment tool, but in our opinion, the negative aspects, such as incorrect assessment or inconsistent scoring, outweigh the strong statistical correlation. If you want to get a "second opinion" on your student's submission, ChatGPT probably will do a good job. Concerning question (RQ-3) "What other programming teaching assistance does ChatGPT offer?" we found out that ChatGP could be used as a personal tutor. When given useful requests ChatGPT is able to modify given examples in order to practice a defined subject and also provides step-by-step solutions, which allow students to review and learn at their own pace. In future work, we plan to investigate whether the

results presented in this paper are also valid for ChatGPT4 and later models and to use the findings of this paper to further investigate the use of AI systems in didactics of programming.

A   Codesnippets and Grading Suggestion

```java
public static boolean search(int[][] arr, int num) {
    for (int i = 0; i < arr.length; i++) {
        for (int j = 0; j < arr[i].length; j++) {
            if (arr[i][j] == num) {
                return true; // if num is found in the array, return true
            }
        }
    }
    return false; // num is not found in the array, return false
}
```

Fig. 5. Solution by ChatGPT which failed one of the CodeRunner test cases.

```java
public static boolean search(int[][] arr, int num) {
    if (arr == null) {
        return false;
    }
    for (int i = 0; i < arr.length; i++) {
        if (arr[i] == null) {
            continue;
        }
        for (int j = 0; j < arr[i].length; j++) {
            if (arr[i][j] == num) {
                return true; // if num is found in the array, return true
            }
        }
    }
    return false; // num is not found in the array, return false
}
```

Fig. 6. Working solution by ChatGPT after a small change request.

Based on the issues identified in the student's solution, I would deduct points as follows:

1. Variable type: Deduct 1 point for using the wrong variable type for `pi`.
2. Division by integer: Deduct 2 points for not using floating-point division in the expression for the Leibniz series.
3. Exponentiation function: Deduct 1 point for not importing the `Math.pow` function.
4. Loop range: Deduct 1 point for using the wrong loop range.
5. Calculation of pi: Deduct 2 points for not dividing the result by `n` before multiplying it by 4.

This results in a total deduction of 7 points. Therefore, the student should receive 3 points out of 10 for this exercise.

Fig. 7. Grading suggestion of ChatGPT

References

1. Abramson, A.: How to use ChatGPT as a learning tool. Monitor Psychol. **54**(3), 36–44 (2023). https://www.apa.org/monitor/2023/06/chatgpt-learning-tool
2. Bender, E.M., Gebru, T., McMillan-Major, A., Shmitchell, S.: On the dangers of stochastic parrots: Can language models be too big? In: Proceedings of the 2021 ACM Conference on Fairness, Accountability, and TransparencyMarch (FAccT 2021), pp. 610–623. Association for Computing Machinery, New York (2021). https://doi.org/10.1145/3442188.3445922
3. Bornat, R., Dehnadi, S., Simon: mental models, consistency and programming aptitude. In: Proceedings of the Tenth Conference on Australasian Computing Education (ACE 2008), vol. 78, pp. 53–61. Australian Computer Society Inc., AUS (2008)
4. Cunningham, K., Blanchard, S., Ericson, B., Guzdial, M.: Using tracing and sketching to solve programming problems: replicating and extending an analysis of what students draw. In: Proceedings of the 2017 ACM Conference on International Computing Education Research (ICER 2017), pp. 164–172. Association for Computing Machinery, New York (2017). https://doi.org/10.1145/3105726.3106190
5. Ericson, B.J., Foley, J.D., Rick, J.: Evaluating the efficiency and effectiveness of adaptive parsons problems. In: Proceedings of the 2018 ACM Conference on International Computing Education Research (ICER 2018), pp. 60–68. Association for Computing Machinery, New York (2018). https://doi.org/10.1145/3230977.3231000
6. Hanks, B., Brandt, M.: Successful and unsuccessful problem solving approaches of novice programmers. In: Proceedings of the 40th ACM Technical Symposium on Computer Science Education (SIGCSE 2009), pp. 24–28. Association for Computing Machinery, New York (2009). https://doi.org/10.1145/1508865.1508876
7. Honegger, B.D.: ChatGPT & School - Assessments by the Chair of "Digitalisierung und Bildung" at the University of Teacher Education Schwyz. pädagogische hochschule schwyz (2023). https://mia.phsz.ch/pub/LLM/WebHome/2023-chat-gpt-und-schule-v128-en.pdf. Accessed 27 Apr 2023

8. Jacques, L.: Teaching cs-101 at the dawn of chatgpt. ACM Inroads **14**(2), 40–46 (2023). https://doi.org/10.1145/3595634

9. Joyner, D.A.: ChatGPT in education: partner or pariah? XRDS **29**(3), 48–51 (2023). https://doi.org/10.1145/3589651

10. Kesselbacher, M., Bollin, A.: Towards the Use of Slice-based cohesion metrics with learning analytics to assess programming skills. In: Third International Workshop on Software Engineering Education for the Next Generation (SEENG). arXiv preprint arXiv:2105.04974 (2021)

11. Leinonen, J., et al.: Using large language models to enhance programming error messages. In: Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023), vol. 1, pp. 563–569. Association for Computing Machinery, New York (2023). https://doi.org/10.1145/3545945.3569770

12. Lister, R., Fidge, C., Teague, D.: Further evidence of a relationship between explaining, tracing and writing skills in introductory programming. SIGCSE Bull. **41**(3), 161–165 (2009). https://doi.org/10.1145/1595496.1562930

13. Lister, R., Leaney, J.: Introductory programming, criterion-referencing, and bloom. ACM SIGCSE Bull. **35**(1), 143–147 (2003). https://doi.org/10.1145/792548.611954

14. Lister, R., Simon, B., Thompson, E., Whalley, J.L., Prasad, C.: Not seeing the forest for the trees: novice programmers and the SOLO taxonomy. ACM SIGCSE Bull. **38**(3), 118–122 (2006). https://doi.org/10.1145/1140123.1140157

15. Ma, L., Ferguson, J., Roper, M., Wood, M.: Investigating and improving the models of programming concepts held by novice programmers. Comput. Sci. Educ. **21**(1), 57–80 (2011)

16. de Raadt, M., Watson, R., Toleman, M.: Teaching and assessing programming strategies explicitly. In: Proceedings of the Eleventh Australasian Conference on Computing Education (ACE 2009) vol. 95, pp. 45–54. Australian Computer Society Inc, AUS (2009)

17. Rich, K.M., Strickland, C., Binkowski, T.A., Moran, C., Franklin, D.: K-8 learning trajectories derived from research literature: sequence, repetition, conditionals. ACM Inroads **9**(1), 46–55 (2018). https://doi.org/10.1145/3183508

18. Sarsa, S., Denny, P., Hellas, A., Leinonen, J.: Automatic generation of programming exercises and code explanations using large language models. In: Proceedings of the 2022 ACM Conference on International Computing Education Research (ICER 2022), vol. 1, pp. 27–43. Association for Computing Machinery, New York (2022). https://doi.org/10.1145/3501385.3543957

19. Snowdon, S., Snowdon, S.: Explaining program code: giving students the answer helps - but only just. In: Proceedings of the Seventh International Workshop on Computing Education Research (ICER 2011), pp. 93–100. Association for Computing Machinery, New York (2011). https://doi.org/10.1145/2016911.2016931

20. Teague, D., Lister, R.: Programming: reading, writing and reversing. In: Proceedings of the 2014 Conference on Innovation and Technology in Computer Science Education (ITiCSE 2014), pp. 285–290. Association for Computing Machinery, New York (2014). https://doi.org/10.1145/2591708.2591712

21. Zhai, X.: ChatGPT user experience: implications for education (2022)

Competitions, Problem Solving, and Computational Thinking

All Green: How Different Age Groups Solved the Same Bebras Task

Carlo Bellettini[1] , Violetta Lonati[1,2] , Mattia Monga[1,2(✉)] ,
and Anna Morpurgo[1,2]

[1] Università Degli Studi di Milano, Milan, Italy
[2] Laboratorio Nazionale CINI 'Informatica e Scuola', Rome, Italy
{bellettini,lonati,monga,morpurgo}@di.unimi.it

Abstract. In this report we investigate how pupils tackle an interactive problem-solving task recently proposed in the Bebras Challenge. The task comes across as fun and simple and we proposed it to all pupils from fourth grade up. We analyzed the data collected, produced by 18,486 participating teams while interacting with the local Bebras platform. The quantitative data were supplemented by interviews conducted after the challenge, in which students were asked to solve the question while thinking aloud. Regardless of their age, all the participants found a good challenge in the task and they spent a lot of time interacting with it. Almost all teams attempted a naive approach to the solution, misled by the superficial characteristics of the problem, and many insisted on attempting the naive approach without ever abandoning it; this behaviour clearly decreases with increasing age.

Keywords: K12 · Bebras challenge · observational study · problem-solving

1 Introduction

The Bebras International Challenge on Informatics and Computational Thinking (http://bebras.org) is a yearly contest organized in several countries since 2004 [1,3], with more than three million participants worldwide. The contest, open to pupils of all school levels (from primary up to upper secondary), is based on tasks rooted on core informatics concepts, yet independent of specific previous knowledge such as for instance that acquired during curricular activities.

Each Bebras country is free to choose and adapt the tasks to the local school context. In most countries the contest is run individually, in others it is team based. Many countries also propose *interactive* tasks, *i.e.,* tasks whose solution requires to interact with the contest platform. In addition to the submitted answers, our Bebras platform [4] is able to collect data concerning the interactions with the platform itself (how much time pupils spend on each specific task, whether and when they go back and review/change their answer to an already

completed task, whether they perform actions that generate feedback from the system, and so on [5,6]). This offered us the chance to conduct an observational study about how pupils of different age groups behave to find a solution. We focused on a task that we thought could be proposed to all pupils from fourth grade up. It is based on a geometric figure; the goal is to transform the figure, through a very simple graphic processing system, by applying a limited number of operations.

Here we present an analysis of the data collected during the Italian contest, produced by 18,486 participating teams while interacting with the Bebras platform. We studied which sequences of operations were performed by teams, in order to identify repeated patterns, successful and unproductive approaches, and changes in strategies. The quantitative data were supplemented by a think-aloud protocol conducted after the challenge, in which we asked five pairs of students to solve the task while thinking aloud and then interviewed them with follow-up questions.

The paper is organized as follows. Section 2 describes the task; in Sect. 3 we report our observations and analyses of the data collected; Sect. 4 summarizes our findings and draws some conclusions.

2   A Fun Task that Is Harder Than It Looks

2.1   The Task

This study focuses on one specific task (2022-UA-01a Filling), authored by the Ukrainian Bebras team, and included in the Bebras Challenge in 2022 by ten countries, who translated and adapted it to the local contexts and platforms. In our country, where the contest involves teams of two or three people, the task was implemented as an interactive task, and it was proposed to all age groups, from IV to XIII grade. Figure 1 shows our version of the task, translated back into English for this paper's readers. The colors were associated to patterns to help colorblind pupils, but our text forgot to explicitly mark which is the "green color" (the solid one): please note, however, that since our contest is team based, it should be considered rather unlikely that all the members of a team are colorblind.

The interactive version we proposed was designed to allow any number of trials; it has a counter of the moves and the solvers could reset to the initial state at will. This allows the teams to explore the system and figure out its semantics, which is described in the task only in very generic terms. In particular, the task's text does not define what a *region* is: any maximal union of connected circles with the same colour (at the beginning, there are seven regions). Similarly, the task's text does not state the fundamental property that solvers need to infer from trials: if one fills a region with the same color of one of its neighbours, then the filled region and its neighbours with the same color are merged into one single region and, from then on, their colors will always change simultaneously.

2022-UA-01a_Filling

The program in the frame allows you to change the colors of the picture.

Select a color by clicking on one of the squares on the right, then click on a region of the picture: this will change the color of the region.

You can try as many times as you want; to restart just push the button "Restart".

You must color **all the picture with the green color**.

The number shows how many times you clicked. Try to reach the goal with the minimum number of clicks: it is possible to do that with just **3** clicks!
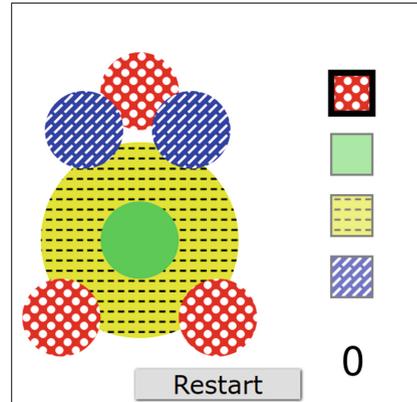
Fig. 1. The "All green" task; 'green' is the only solid color. (Color figure online)

2.2   The Task's Solution

The figure can be turned into green by filling each circle with green separately; this requires six moves. By exploiting the possibility to merge circles and change their colors simultaneously, the number of moves can be reduced down to three. Here is the shortest sequence of moves leading to the desired result: fill with the blue color the largest circle (you now have five regions); fill the resulting blue region with red (you now have two regions); fill the resulting red region with green (you now have one single green region).

Such a sequence is unique, as can be proved as follows. The initial figure has four colors and three of them must be dropped. Since the number of allowed moves is three and no more than one color can be dropped in one move, then one color must disappear at each move. In the beginning, the only color that can be dropped in one single move is yellow, so one must click on the yellow circle to turn it into either red or blue. In the former case, both the red and the blue would appear in two separate regions, so neither of them could be removed in the next move. Thus, one must turn the yellow circle into blue. The next move is forced, since filling the resulting blue region with red is the only move that can reduce the number of colours, and the last one is then obvious.

The above proof is quite straightforward, but it needs the solver to understand the goal as "remove all colors except green from the picture". This, however, encompasses a totally different mental *representation* of the problem, which the original wording of the task does not evoke.

2.3   How Hard Is the Task?

Bebras authors marked the task as one of medium difficulty for the age group IV-V grade, but we felt it could be proposed to all the age groups: the task comes across as fun and simple at all ages, but could be challenging enough even for grown-ups. As reported in Table 1, the task in the version used in our country turned out to be more difficult than estimated, especially for the youngest pupils. Less than 10% of the primary school teams were able to solve the task; the success rate increases with the age, regularly and markedly, but it remains at the level of challenging tasks even for the higher school grades.

Table 1. Overall results

grade	teams	success %
IV–V	3,008	8.9
VI–VII	8,001	25.0
VIII	3,648	35.5
IX–X	2,074	50.3
XI–XIII	1,755	64.8
Total	18,486	31.1

3   Observations

The data collected by the Bebras platform concern the interactions of teams with the platform itself, and in particular which actions they performed and when they clicked on the "Restart" button to recover the original figure and reset the counter for the number of moves. Before presenting our findings, let us introduce some preliminary definitions.

A *round* is a sequence of moves delimited by two Restart moves (obtained by one of the following actions: click on the Restart button, or open the task for the first time, or leave the task for the last time). The *length of a round* is the number of its moves. When the figure is all green, we say that it is in the *all-green configuration*; from this configuration on, the color of the figure can change but will remain uniform. A *green round* is a round that ends in the all-green configuration; there is only one *successful* green round of length 3. An *ineffective move* does not change the figure; ineffective moves are obtained whenever one clicks on a region after having selected the same color as the region (they are counted by the system as any other move). A *green-move* (resp., blue-move, or red-move) is a move that turns some region into green (resp., blue, or red).

The data collected during the context allow us to reconstruct the behaviour of each team. For instance, Fig. 2 illustrates the behaviour of an VIII grade
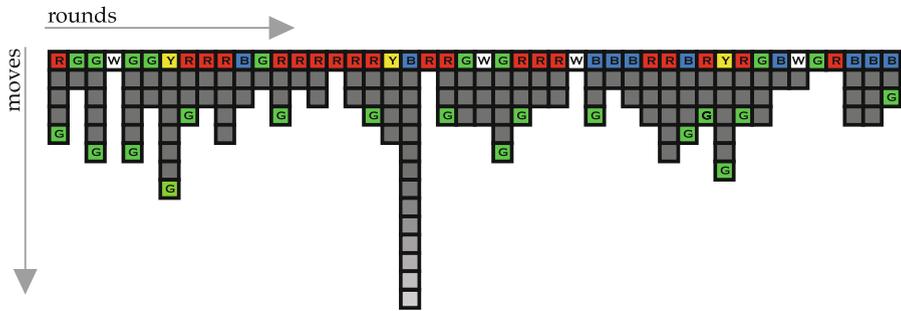
Fig. 2. Diagram summarizing the behavior of a (successful) team (VIII grade).

team who succeeded in solving the task. Each square represents a move, and each column represents a round, the leftmost being the first round carried out. The top square in each column is colored with the color applied in the first move (white, if the move is ineffective), the bottom square in each column is colored in green if the column represents a green-round. This particular team pressed the Restart button 45 times, hence producing 46 rounds. There are 16 green rounds, and the last one has length three. Four rounds start with an ineffective move, two among which followed by a restart, hence producing "empty" rounds.

All this team's rounds are short, except for the 20th (the one fading towards the bottom of the figure). In general, rounds that either are longer than six moves or contain more than one ineffective move, can be considered as exploratory or with no intentional purpose, since i) six is the number of moves required to make the figure all green proceeding circle-by-circle and ii) an ineffective move may occur by mistake, but more than that is not reasonable in a serious attempt, considering the limit of three moves overall. Hence, we will call *trials* only the rounds that are six-moves long at most and contain one ineffective move at most.

3.1   Engagement

Teams spent a lot of time on the "All green" task. The overall time allowed in the context was 45 min per 12 tasks (or, for the youngest, 10 tasks), that is less than 4 min for each task on average. As reported in Table 2, teams spent, on average, 7 min and a half on the task (about 19% of the whole time) with peaks occurring on the central age levels. The pattern for the teams who solved the task, however, is slightly different: the younger successful solvers spent *more* time than average on the task, while the older spent less time. The time spent varies a lot among teams, even within a single category, as shown by Fig. 3, where we use kernel density estimation (instead of the usual histograms) for readability and visual comparison.

The active and positive engagement of participants was clearly visible also in the think-aloud protocol: participants spent several minutes on the task, and the frustration that occasionally emerged from the difficulty of finding the correct

Table 2. Cumulative time spent on the task (teams could leave the task and go back to it later, as many times as desired)

grade	time spent on the task	fraction of total time	avg n. of rounds	avg n. of trials	time spent on the task	fraction of total time	avg n. of rounds	avg n. of trials
	all teams				*successful teams*			
IV-V	443 s (7.4 m)	20.3%	18.8	25.3	574 s (9.6 m)	25.5%	16.3	22.0
VI-VII	504 s (8.4 m)	21.7%	25.0	26.9	534 s (8.9 m)	23.2%	21.5	23.1
VIII	456 s (7.6 m)	18.6%	24.7	23.8	438 s (7.3 m)	18.2%	21.2	20.7
IX-X	357 s (6.0 m)	14.4%	20.0	18.6	333 s (5.5 m)	13.4%	17.7	16.4
XI-XIII	310 s (5.2 m)	12.4%	17.0	15.0	278 s (4.6 m)	11.3%	15.3	13.4
Average	450 s (7.5 m)	19.2%	22.6	22.3	427 s (7.1 m)	18.0%	19.6	19.4

solution generally did not extinguish the motivation to keep trying. Only one group decided to give up, settling for a 4-move solution.
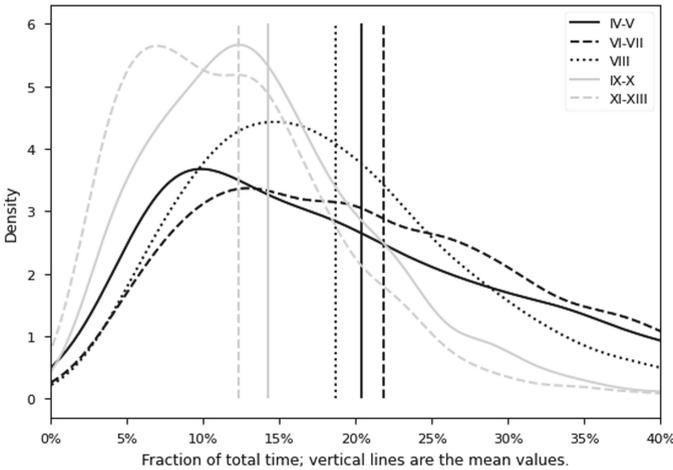
Fig. 3. Fraction of total time spent on the task by all teams (kernel density estimation plots).

Another measure of engagement is provided by the number of rounds and trials carried out by teams: the averages by category are also reported in Table 2. In all age categories, the average number of trials and rounds is at least 15. The pattern is similar to the one for the time spent on the task: for the lowest two categories, the number of attempts increases when considering only the successful teams, while for the other ones it decreases. We interpret this as follows: for the younger ones, more attempts help to succeed and persistence seems useful; for the higher school levels, fewer attempts are needed to succeed; for the central grades, success does not seem immediately connected with persistence.

3.2   Graphs of Moves

To analyse the frequencies of moves in rounds and trials, we consider the graph of all possible configurations reached by one team. Such graph is defined as follows: the nodes of the graph are all possible configurations of the figure (*i.e.,* all possible ways the figure can be coloured); the arcs represent the possibility to switch from one configuration to another with a single click. Both the nodes and the arcs are equipped with weights that denote respectively in how many rounds/trials of the team a configuration (node) was reached and how many times a certain move (arc) was performed.

Table 3 reports the average number of nodes of such graphs, over the same age category. The number of reached configurations is lower when considering only trials, which also depends on the fact that fewer configurations are within reach with only 6 moves. The numbers in the table show a pattern similar to the one in Table 2.

Table 3. Average number of explored configurations, in rounds and in trials.

grade	conf. in rounds	conf. in trials	conf. in rounds	conf. in trials
	all teams		*successful teams*	
IV–V	31.05	25.91	36.95	32.45
VI–VII	37.54	31.99	38.53	33.34
VIII	37.29	32.45	35.26	30.87
IX–X	32.01	28.21	29.31	26.14
XI–XIII	27.32	24.82	24.38	22.15

The graph of moves helps also identify which configurations occur the most and which are the most probable sequences of moves. For instance, Fig. 4 shows the graph of moves for all VIII grade teams. The weights of configurations and transitions are represented by the nodes' background and arcs' darkness. The figure includes only the nodes representing the most recurring configurations, namely those whose frequency is more than 4% of the most frequent configuration (which is the starting one, obtained after each restart). Moreover, to avoid dispersion of data, we merged the symmetrical nodes, that is, those that represent symmetrical pictures (*e.g.,* we basically consider equivalent filling with blue the bottom-left or the bottom-right circle). One can see that, from the initial configuration, there are four arcs towards configurations where one of the little circles is changed (the four arcs weigh 18% all together), and three arcs towards configurations where the big central circle is changed into blue, red, or green respectively (from left to right). They collect respectively 14%, 19% and 10% of all the choices.

During the think-aloud protocol we noticed in particular that many different configurations were obtained with green-moves by the youngest pupils who did
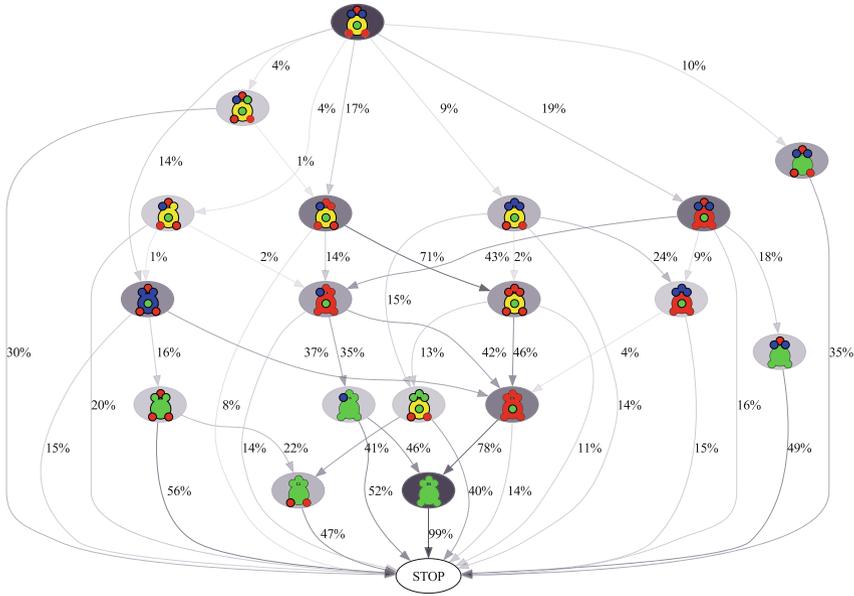
Fig. 4. Graph of the configurations most frequently reached by teams of VIII grades.eps

not find the solution, as they tried in many different ways to turn the figure into green by only using green-moves. Such configurations were not visited by the older students who found the solution, as they realized sooner that the green-moves were not fruitful, given the limit of three moves. The frequency of such configurations is indeed low also in the graph of Fig. 4.

3.3   Strategies

To analyze the strategies adopted by teams, we looked at the evolution of trials, by focusing on the first move of each trial. In all categories, one can see that a large proportion of the first few trials starts with a green-move; initial red-moves prevail in the central rounds, initial blue-moves occur more rarely and increase towards the final rounds when considering only successful teams.

This is consistent with the strategy that we observed in the think-aloud protocol. Except for one group who solved the task very quickly, the other two groups who found the solution reasoned as follows: at first they tried to turn the whole figure into green by making green one circle at a time, hence the first few trials started with a green-move. After a while they realized that, with the above strategy, too many moves are needed and consequently switched to using different initial colours than green. Red is the most frequent colour in the original figure and this led them to try turning the whole figure into red instead of green. However this was not the successful strategy; indeed, after some trials,

they understood that the desired result could not be achieved if starting with a red-move, and concluded that they should start with a blue-move. Yet, this was not enough to find immediately the successful 3-long sequence of moves, which required some further attempts, with some revival attempts starting again with a red-move.

This approach can be seen also in the data collected by the platform. Figure 5(a) illustrates it by contrasting the occurrences of initial green/red/blue moves for VIII grade teams. The left and right portions of the figure were constructed using data exclusively from the unsuccessful and successful teams, respectively. Each team is depicted as a row. Rows are sorted from top to bottom according to the ratio of initial green-moves (resp., red/blue-moves) over the number of trials. At the top of the leftmost diagram, one can notice the relevant portion of teams who start with a green-move in most of their trials. At the top of the rightmost diagram, one can notice a small portion of teams who start with a blue-move in most of their (few) trials.

Figure 5(b) contrasts the behaviour of teams among the school levels. To simplify the figure, we only show whether the initial moves of the rounds are green or not. As in the Fig. 5(a), each row represents a team, and the teams are sorted according to the longest prefix of trials that start with a green-move. The five diagrams are scaled so that they have the same height, even if they represent populations of different sizes. This allows the reader to perceive how the percentiles change: moving towards right, there is an increase in the number of teams who abandon the green-strategy early.

unsuccessful teams   successful teams   IV - V   VI - VII   VIII   IX - X   XI - XIII

(a)   (b)
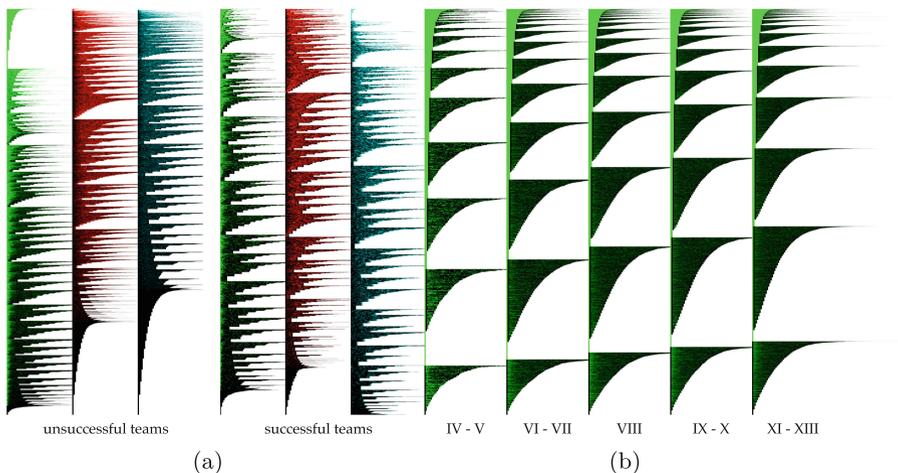
Fig. 5. (a) Colors of initial moves, with unsuccessful/successful teams on left/right, respectively. (b) Trials starting with green-moves, grouped by category. (Color figure online)

3.4   Statistical Relation Between Success and Overall Performance

In order to study the statistical relationship between the probability of success in the task and the overall performances in the challenge, we considered the score each team gained in all the other tasks (*i.e.,* the "All green" task excluded). The scores were standardized to make them comparable among different categories: each score s was mapped to a standardized score $\frac{s-mean_s}{stdev_s}$, therefore having a mean score map to a standardized 0.0 in all the categories, and standardized scores ranges from –2.96 to 3.22. Then we fitted a generalized linear model to measure the effect of score on the probability of success, stratified by category. The model is the following, where β_K and β_S are two vectors of five parameters to be fitted (one for each category), which respectively measure the effect of being a team in a specific category and having performed with a standardized score S (see [2] for further details on this approach):

$$\begin{aligned}
\beta_K &\sim Normal(0, 0.5) \\
\beta_S &\sim Normal(0, 0.5) \\
p_i &= logit^{-1}(\beta_K + \beta_S \cdot S) \\
y_i &\sim Bernoulli(p_i)
\end{aligned} \tag{1}$$

As shown in Table 4, the teams who performed better overall (*i.e.,* those with standardized score 1.0), had a higher probability of getting the "All green" task right than those who have an overall average performance (*i.e.,* those with standardized score 0.0). The increment, however, is smaller for primary school teams.

Table 4. Parameters estimated by the Generalized Linear Model (1) (mean values of Monte-Carlo Markov Chains simulations)

grade	mean β_K	mean β_S	probability of success with standardized score 0.0	probability of success with standardized score 1.0	increment for better performers
IV-V	—2.5	0.7	7.7%	14.6%	6.8%
VI-VII	–1.6	0.5	24.0%	39.1%	15.1%
VIII	–0.6	0.5	34.9%	52.2%	17.3%
IX-X	0.0	0.4	50.5%	67.5%	17.0%
XI-XIII	0.7	0.6	65.9%	79.7%	13.8%

4   Conclusions

In this report we described the observations we did on how different age groups tackled the same interactive problem-solving Bebras task. Regardless of their age, all the participants found a good challenge in the task, as shown by the

time spent interacting with it: overall almost $\frac{1}{5}$ of the contest time was spent on the task, leaving only $\frac{4}{5}$ for the remaining 9–11 tasks. Our analyses show that the ability to solve the task increases with age (as expected), regularly and markedly; for primary school pupils (in fact the age group targeted by the original authors of the task) the task turned out to be very difficult, and all the data suggest that this age group had the greatest difficulties in planning a winning strategy overcoming a failing naive approach. Except for the small minority of teams who found the solution quickly, most teams (even considering only those who succeeded in solving the task) carried out many trials. Initially, almost all teams attempted a naive approach, misled by the superficial characteristics of the problem, and many insisted on attempting the naive approach without ever abandoning it; this behaviour clearly decreases with increasing age. Moreover, while for younger kids a successful solution is associated with a higher number of visited configurations, for older ones the pattern is reversed: at some point in the process there is possibly an *eureka* moment in which they grasp a new (more productive) way of representing the problem in their mind. In this paper we mainly described the results derived from the analysis of quantitative data, with some further insight obtained by some interviews with subjects requested to try to solve the "All green" task while thinking aloud. Overall, we believe this kind of study is important to improve the design of Bebras tasks and a general understanding of interactive problem-solving.

References

1. Dagienė, V.: Sustaining informatics education by contests. In: Hromkovič, J., Královič, R., Vahrenhold, J. (eds.) ISSEP 2010. LNCS, vol. 5941, pp. 1–12. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11376-5_1
2. Gelman, A., Hill, J., Vehtari, A.: Regression and other stories. Cambridge University Press, Cambridge (2020)
3. Haberman, B., Cohen, A., Dagienė, V.: The beaver contest: attracting youngsters to study computing. In: Proceedings of ITiCSE 2011, pp. 378–378. ACM, Darmstadt, Germany (2011)
4. Bellettini, C., et al.: A platform for the Italian Bebras. In: Proceedings of the 10th International Conference on Computer Supported Education (CSEDU 2018), vol. 1, pp. 350–357. SCITEPRESS, March 2018. https://doi.org/10.5220/0006775103500357
5. Bellettini, C., Lonati, V., Monga, M., Morpurgo, A.: How pupils solve online problems: an analytical view. In: Proceedings of the 11th International Conference on Computer Supported Education (CSEDU 2019), vol. 2, pp. 132–139. SCITEPRESS, May 2019. https://doi.org/10.5220/0007765801320139
6. Bellettini, C., Lonati, V., Monga, M., Morpurgo, A.: Behind the shoulders of bebras teams: analyzing how they interact with the platform to solve tasks. In: Lane, H.C., Zvacek, S., Uhomoibhi, J. (eds.) CSEDU 2019. CCIS, vol. 1220, pp. 191–210. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58459-7_10

Effects of the COVID-19 Pandemic on the Bebras Computational Thinking Challenge: Comparing Numbers, Examining Reasons and Investigating Recommendations

Martin Kandlhofer[1]([✉]) [iD], Wilfried Baumann[1] [iD], Gerald Futschek[2] [iD], Liam Baumann[1] [iD], and Steven Ludwig[1] [iD]

[1] Austrian Computer Society OCG, Vienna, Austria
{martin.kandlhofer,wilfried.baumann,liam.baumann,steven.ludwig}@ocg.at
[2] Vienna University of Technology, Vienna, Austria
gerald.futschek@tuwien.ac.at

Abstract. The Bebras Challenge is organized in more than 70 countries worldwide. It aims to foster pupils' interest and passion for informatics and computational thinking. Although the Bebras Challenge is by its nature an online activity, most countries had a decline in participation numbers compared to non-pandemic years. Some countries recorded larger relative declines than others, certain countries even recorded increases. In order to investigate this issue, a comprehensive study, applying quantitative and qualitative methods, was conducted. Based on reported participation data, an interactive online graphic was created in which the reach (participation per thousand inhabitants) of the Bebras Challenge in individual countries or groups of countries in the respective years can be compared with each other. Following an online research regarding homeschooling during the Bebras weeks in different countries, an online survey among 40 Bebras representatives worldwide delivered important key data, such as main reasons for an incline or decline of participation numbers or which measures have been taken to hold the challenge during the pandemic. This was accompanied by qualitative interviews with selected Bebras representatives. The results of this study could help organizers of national and international school activities to respond more effectively to possible future adverse situations.

Keywords: Bebras Challenge · Computational Thinking and Informatics · Pandemic Impact

1 Introduction

As digitization steadily increases, ICT becomes an integral part of our everyday lives. It is therefore crucial to foster skills which young people need to ensure

them a social and economic participation in this rapidly changing technological world. Having knowledge of fundamental informatics concepts and being able to perform computational thinking (problem-solving, algorithmic-thinking, logical-reasoning) are key skills of the 21st century [6,17]. Introducing young people to informatics and computational thinking in a playful and engaging way can help to develop a strong foundation in this area.

In this context, the international Bebras Challenge on Informatics and Computational Thinking has proved to be extremely effective. With a yearly participation of more than 3,000,000 school students from more than 70 countries, the Bebras Challenge is the largest school competition in the area of informatics [2]. The international recommendations and initiatives to increase the amount of teaching hours related to digital skills and informatics at school led to steadily increasing participation numbers in most of the participating countries. The Bebras Challenge is performed online and usually at school under surveillance of teachers during one or two weeks in November [3,4]. Since the Bebras Challenge is an online competition, it had a good chance to be taken by the students during lockdown and homeschooling.

For investigating the effects of the COVID-19 pandemic on the Bebras Computational Thinking Challenge, this work was guided by the following three main research questions:

– Q_1: Did homeschooling have an influence on the participation numbers of the Bebras Challenge in different countries?
– Q_2: What were the reasons for a decrease or an increase of participation numbers?
– Q_3: What are possible recommendations for Bebras organizers to better address future adverse situations?

The remainder of this paper is structured as follows: Sect. 2 summarizes related literature, Sect. 3 outlines the applied methodology, Sect. 4 presents the results of our investigation, Sect. 5 discusses the findings with regard to the guiding questions, while Sect. 6 provides conclusions, limitations and outlooks.

2   Related Research

Only very few studies have explored shifts taking place within participation rates for the Bebras Challenge for specific regions, yet none so far have examined these trends on a larger scale across wider geographical areas such as Europe. An analysis recently conducted by Maranatha Bebras Bureau Christian University offers unique insights into this matter regarding the region of Indonesia. Despite facing numerous challenges posed by distance learning during the COVID-19 pandemic period last year, they recorded over twice as many students participating at their Bebras Bureau in 2020 compared with numbers just one year before (i.e., increasing from only 6,846 students in 2019 to 16,177 in 2020). The investigation also highlighted the apparent effectiveness of teacher workshops and students' enthusiastic participation to ensure the success of these outcomes, despite the various obstacles posed by remote learning during the pandemic [1].

A similar initiative, the Math Kangaroo - a highly regarded international mathematics competition for school students - usually takes place on the third Thursday of March and usually uses paper and pencil multiple choice tests. They had a significant decline of participation numbers in the years 2020 and 2021 after a long period of steadily increasing participation numbers. The competition had over 6 million participants from 57 countries in 2014, and by 2022, it had 84 participating countries and claimed to be the largest competition for school students in the world [8]. In the United States, participation in the competition was reported 30,550 students in 2022 and 31,004 in 2021 which was a decrease from 35,171 participants in 2020. However, in 2023, the competition saw a significant increase with 36,421 participants [12]. Compared to the figures from the US, the figures from the Netherlands and Germany show a much higher decline [9,10].

The pandemic has led to a significant surge of online education, which is discussed in Sá and Serpa's paper "COVID-19 and the Promotion of Digital Competences in Education." They argue that schools require robust educational infrastructure to ensure that remote teaching can be successful during such a pandemic. To support effective online instruction, teachers must adapt and develop standardized home-based equipment while developing crucial digital competencies. This paper discusses, namely, which changes were made and how. Sá and Serpa argue for national-level research on remote-learning methods and practices as well, which can help educators implement effective training methodology. They also acknowledge challenges associated with disparities in access to digital tools, worsened by the pandemic impact at large. While the document provides a broad understanding of the impact of the pandemic on digital learning, it lacks concrete data on how the pandemic has specifically altered participation numbers in IT competitions [14]. Our study targets participation numbers exclusively within one of the biggest, namely the Bebras Challenge, during these times when concrete data has been reported less frequently despite requiring the attention from an overall research standpoint.

It's important to recognize that several factors can affect participation rates during the pandemic. These reasons are contingent upon various elements such as the manner in which schools were impacted by the pandemic, lockdown types/durations, homeschooling frequency along with other such variables. Nevertheless, there seems to be a need for comprehensive studies that allow us to better comprehend patterns and factors influencing these statistics across different areas/cultures. This leaves a gap in our understanding of the global impact of the pandemic on student engagement in computational thinking challenges, like Bebras. The goal of this research paper is to close this gap, enable further research in those topics, and spark discussions.

3   Methodology

Qualitative and quantitative methods have been applied to gather data with regard to the main research questions (which were stated in Sect. 1) [13], following an inductive approach [15]. Collected data has been anonymized and treated

confidentially in accordance with data protection regulations. The next paragraph describes the steps and methods used in this process, while results of each step are presented and discussed in Sects. 4 and 5.

a) Participation numbers: The first step was the collection of participation numbers of past Bebras Challenges based on the reported participation numbers of the community. On the basis of this data, an interactive visualization was created. This visualization provided further insights into the data set and also eased the task of data cleansing, which was done in close cooperation with the national Bebras representatives (e.g. in case of ambiguities, open questions or potential errors in the data).
b) Homeschooling: The next step was to perform an online research regarding type and length of homeschoolings based on the data of the *European Centre for Disease Prevention and Control*[1] [5,11]. This information was then correlated with the participation numbers, taking into account the slightly different dates for the Bebras Challenge in each country.
c) Survey: Based on the findings of the previous steps, an online questionnaire[2] was created and distributed among the Bebras community worldwide using *Google Forms* [16]. The survey comprised questions regarding measures to ease participation during the pandemic, reasons why numbers declined or inclined, as well as further comments or suggestions. Furthermore, a link to the interactive visualization was included in order to additionally verify the participation numbers investigated in step a). The survey also served the purpose of further investigating reasons for an incline or decline of numbers of certain countries (which were identified by the analysis performed in step b).
d) Interviews: Additionally, semi-structured interviews [7] with selected Bebras representatives were prepared and conducted. In particular, countries which have recorded small or large decreases/increases during the pandemic years were of interest. The goal of the interviews was to get more in-depth information regarding the reasons for an increase or decrease, and to examine good-practice examples for handling this difficult situation.

4   Results

By adhering to the methodology outlined in the previous section, the following results could be acquired.

Step a: Participation data from 2004 to 2022 from 57 countries worldwide was collected and visualized as interactive online graphic (Fig. 1)[3]. The visualization shows the reach (participation per thousand inhabitants) of the Bebras Challenge for individual countries or groups of countries in the respective years.

---

[1] Here the focus was on EU countries.
[2] https://forms.gle/rdzcFLbnpzanyg3PA.
[3] The decline of Belarus' and Russia's numbers are a result of being suspended from the Bebras Challenge.

This could also be useful for the community, e.g. for comparisons between countries of similar size or between neighboring countries.
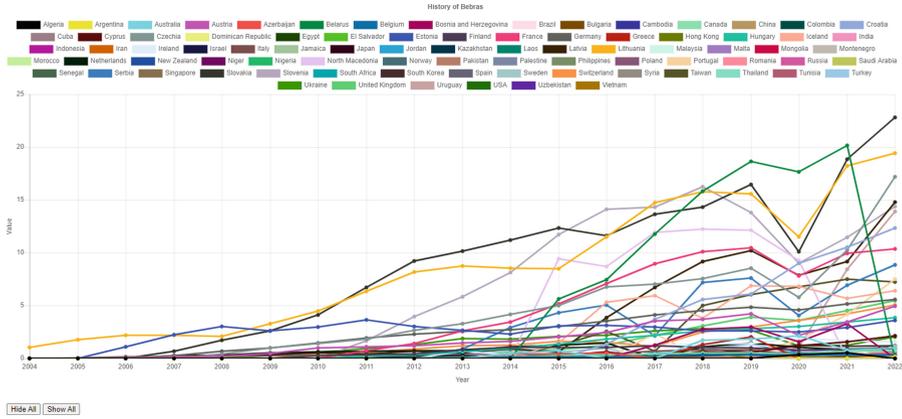

Fig. 1. Visualization of Bebras participation numbers as interactive online graphic (screenshot); the y-axis shows the reach (participation per 1000 inhabitants; or in other words: from 1000 people - how many took part in Bebras in that year); the graphic is available at https://www.coding4you.at/history_of_bebras/

Step b: The analysis of the data clearly showed a correlation between homeschooling and a decline of participation numbers (Fig. 2). In general, most countries that experienced a collision of homeschooling and the Bebras week demonstrated a decline of participation numbers, whereas most countries where the Bebras week did not collide with homeschooling accordingly did not show a decline of numbers (Fig. 3). Exceptions from these observations could also be identified. For instance, Hungary showed an increase of participation numbers during homeschooling, while, for instance, Bulgaria showed a decrease of numbers, although homeschooling did not overlap with the Bebras week. Reasons for this were investigated in the subsequent steps and are discussed in Sect. 5.

Step c: The response rate was high - in total 40 representatives from 35 different countries participated in the online survey[4]. Sixty percent of the survey respondents reported a **decline of participation numbers** during the pandemic, stating the additional overall workload as well as lockdown/homeschooling during the Bebras week as the main reasons for a decline of numbers (Fig. 4). Further factors that led to a decline include the following:

- Lack of suitable devices/technology and/or lack of proper internet access at home.
- Access to computer room was sometimes limited, due to regulations put in place by the government, which forced some schools to allocate computer rooms for other purposes.

---

[4] Multiple answers per country were allowed.
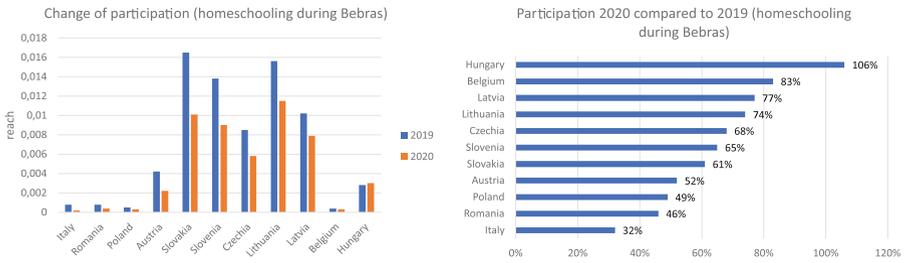
Fig. 2. Countries in which Bebras week collided with homeschooling: Change of reach (participation per thousand inhabitants) from 2019–2020 (left); participation numbers compared between 2019 and 2020, for instance, in 2020 Italy only had 32% the numbers of 2019 (decline), while Hungary had 106% the numbers of 2019 (incline)
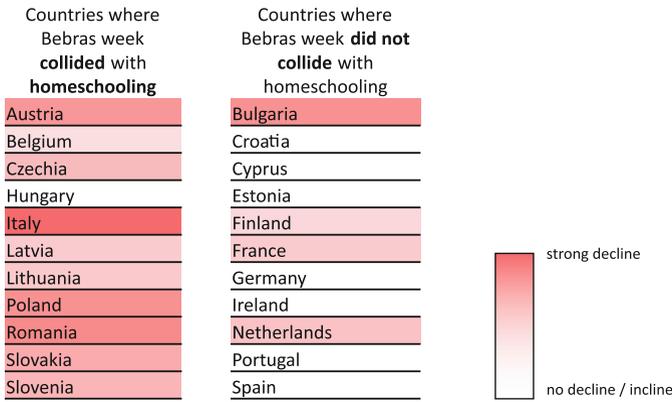
Fig. 3. Correlation between homeschooling and decline of participation numbers based on reach (participation per thousand inhabitants) from 2019–2020.

– In general, the government support for schools experienced a decline in certain countries. Furthermore, in several other countries, the government cancelled school competitions such as Bebras in particular.
– Many teachers reported being very exhausted because many administrations expected the same or even a higher performance of the students compared to regular school years.

In contrast, the survey respondents also reported reasons why Bebras **participation numbers did not decline**, (or even inclined) during the pandemic. A summary of these reasons is presented below:

– Due to fewer opportunities for co-curricular activities, the demand for participation in the Bebras Challenge increased.
– Offering the possibility of participating from home/from anywhere, which encouraged even those students who might not have participated in school.
– Most students already participated from home even before the pandemic.

– Certain countries did not close schools for as long and extensively as other countries during the pandemic.
– Students were well-equipped to engage online from home, with computers and internet connections readily available.
– The Bebras Challenge was proposed by one country as part of a nationwide computational thinking program, which saw an increase in participation year by year.
– A large campaign was conducted through schools to promote participation in the Bebras Challenge.

In order to **ease students' participation in the Bebras Challenge**, participation from home, and at any time respectively, was encouraged by the majority of the countries (Fig. 5). Other measures mentioned by the study respondents can be summarized as follows:

– Offering flexible dates or extending the Bebras Challenge up to several weeks.
– Postponing the Bebras Challenge until schools reopened.
– Making the Bebras system/platform accessible for the whole Bebras week 24/7, with access from anywhere, respectively.
– Offering the competition at the same time for all students.
– Using old tasks to create "Bebras at home" challenges as practice and as tools for teachers to easily teach computational thinking in remote settings.
– Reopening the access to the Bebras Challenge for the youngest categories so that they could compete from schools after the top of the pandemic wave was over.

Step d: At the time of writing this paper, two interviews with Bebras representatives from Hungary and Switzerland were conducted. Additionally, an interview with a representative of Uruguay was conducted in written form (due to scheduling reasons via email). The responses were qualitatively analyzed, following an inductive approach [15]. The findings of this analysis can be summarized as follows:

The registration of students for the Bebras Challenge was conducted prior to the start of homeschooling. Due to remote teaching limitations faced by teachers of various (non-technical) subjects, the Bebras Challenge was more flexible in its implementation, not being limited to the informatics subject only. During the Bebras Challenge, students were not closely monitored, nevertheless, cases of cheating were minimal. This can be attributed to the high motivation of students and a reduced number of students refusing to participate or submitting incomplete responses. Additionally, some teachers included the Bebras results into their grading, providing an additional incentive for students. A fixed time window was provided for students to complete the Challenge (since a fixed time window might foster students' preparedness to participate). Other solutions included flexible time windows as well as fixed time windows with one alternative time window, ensuring that students had the opportunity to participate at a time that suited best for them. Furthermore, fostering the participation in the Bebras Challenge as part of a nationwide computational thinking program turned out to be a viable solution.

Did Bebras participation numbers decline in your country in any year during the pandemic?

Main reasons for the decline of participation numbers in your country?
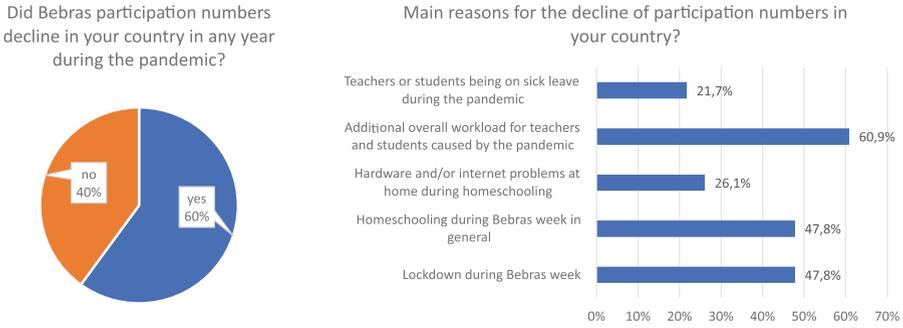
Fig. 4. Did participation numbers decline, and what were the main reasons?

Students could participate from home but at the same time with the rest of the class?

Students could participate from home and at any time?
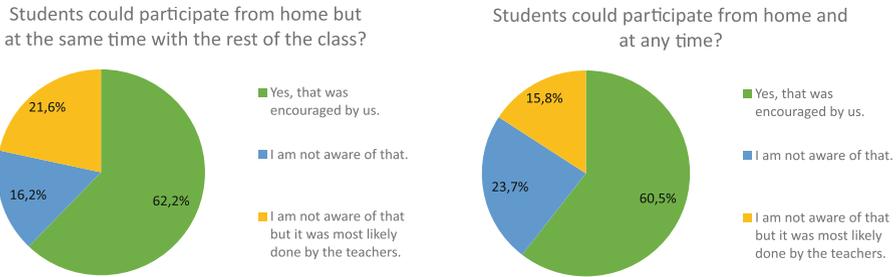
Fig. 5. Measures to ease students' participation in the Bebras Challenge.

5   Discussion

Based on these results, we were able to answer the research questions which were stated in the introduction in Sect. 1.

Regarding Q_1, the gathered and analyzed data showed that, homeschooling had a profound impact on participation numbers of the Bebras Challenge on a worldwide scale. In general, a clear correlation between homeschooling and a decline of participation numbers[5] could be observed (this also corresponds with the numbers of the Math Kangaroo, discussed in Sect. 2). This insight, though not surprising, is based on solid facts through this study. Furthermore, outliers (exceptions) could be observed: Some countries (e.g. Hungary) were able to increase their participation numbers. Their measures taken during the pandemic could serve as best-practice examples.

Regarding Q_2, several factors and reasons for a decrease or an increase of participation numbers could be identified. As presented in Sect. 4, most countries reported a decrease. The main reasons were an additional overall workload for teachers and students during the pandemic, homeschooling/lockdowns during the Bebras week, as well as hardware and/or internet problems at home. Some countries also reported an increase of participation numbers. This was due

---

[5] In this context, it is important to mention that also a number of other circumstances - besides homeschooling - could have affected the participation numbers.

to fewer opportunities for other extra-curricular activities, so the demand for participation in the Bebras Challenge increased. Furthermore, the opportunity to participate from home or from anywhere, attracted students who might not have participated in the regular form. The duration of school closures varied among countries, with some countries experiencing shorter and less extensive closures. In some countries, students were well-prepared for online engagement from home, having access to computers and internet connections. One country also reported that the Bebras Challenge was part of a nationwide computational thinking program. Another country reported that some teachers included the Bebras results into their grading, providing an additional incentive for students to participate.

Regarding Q_3, a number of possible recommendations for Bebras organizers to better address future adverse situations and measures to ease students' participation were already presented in Sect. 4. This included, among others, the possibility to participate from home, to offer flexible time windows (including alternative time windows) for participation, to take into account Bebras results in the school grading system as well as using the opportunity to incorporate Bebras also in non-technical subjects.

6   Conclusion

This paper presented motivation, methods and results of a comprehensive study investigating the effects of the COVID-19 pandemic on the Bebras Computational Thinking Challenge. Guided by three main research questions, it showed - based on solid quantitative and qualitative data - that the pandemic had a profound influence on the participation numbers. It also investigated reasons for a decrease or an increase of participation numbers during the pandemic and presented possible recommendations for Bebras organizers to better address future adverse situations.

The study also comprises certain limitations. For instance, the online survey mainly addressed national Bebras representatives. In this context, more interviews with further national organizers have to be conducted to gain a deeper understanding. It would also be valuable to expand the survey to include teachers who implement the Bebras Challenge at their school. Furthermore, the analyses of homeschooling and participation rates mainly focused on EU countries in the years 2019 and 2020. An upcoming study could also investigate additional countries and years.

The next steps include the further analysis of the gathered data. Since the data set is quite extensive, the presented and discussed results represent only the first findings, consequently, further conclusions can and will be drawn and will also be provided to the Bebras community. Additionally, we will conduct further interviews and continue to collect the participation numbers to keep the interactive online graphic presented in Sect. 4 up-to-date.

Overall, we envision that the work presented in this paper will support organizers of national and international school activities and competitions, similar to the Bebras Challenge, in tackling possible future adverse situations.

References

1. Ayub, M., et al.: Service learning in teachers and students mentoring for 2020 Bebras challenge in pandemic era at maranatha Bebras Bureau Christian University. J. Innovat. Commun. Engag. **2**(2), 75–88 (2021)
2. Bebras Challenge on Informatics and Computational Thinking. https://www.bebras.org/countries.html. Accessed 26 July 2023
3. Dagienė, V., Futschek, G.: Bebras international contest on informatics and computer literacy: criteria for good tasks. In: Mittermeir, R.T., Sysło, M.M. (eds.) ISSEP 2008. LNCS, vol. 5090, pp. 19–30. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-69924-8_2
4. Dagienė, V., Sentance, S.: It's computational thinking! Bebras tasks in the curriculum. In: Brodnik, A., Tort, F. (eds.) ISSEP 2016. LNCS, vol. 9973, pp. 28–39. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46747-4_3
5. European Centre for Disease Prevention and Control. Data on country response measures to COVID-19. https://www.ecdc.europa.eu/en/publications-data/download-data-response-measures-covid-19. Accessed 6 June 2023
6. Geisinger, K.F.: 21st century skills: what are they and how do we assess them? Appl. Measur. Educ. **29**(4), 245–249 (2016)
7. Hove, S.E., Anda, B.: Experiences from conducting semi-structured interviews in empirical software engineering research. In: IEEE International Software Metrics Symposium (2005). https://doi.org/10.1109/METRICS.2005.24
8. Association Kangourou sans Frontières. https://www.aksf.org/index.xhtml. Accessed 8 June 2023
9. Känguru der Mathematik. https://www.mathe-kaenguru.de/chronik/index.html. Accessed 6 June 2023
10. Känguru der Mathematik International. https://www.mathe-kaenguru.de/international/index.html. Accessed 6 June 2023
11. Lionello, L., et al.: Non-pharmaceutical interventions in response to the COVID-19 pandemic in 30 European countries: the ECDC-JRC Response Measures Database. Eurosurveillance **27**(41), 2101190 (2022)
12. Math Kangaroo USA - International Math Competition. https://mathkangaroo.org/mks/. Accessed 1 June 2023
13. Morgan, D.L.: Practical strategies for combining qualitative and quantitative methods: applications to health research. Qualit. Health Res. **8**(3), 362–376 (1998)
14. Sá, M.J., Serpa, S.: COVID-19 and the promotion of digital competences in education. Univ. J. Educ. Res. **8**(10), 4520–4528 (2020)
15. Thomas, D.R.: A general inductive approach for analyzing qualitative evaluation data. Am. J. Eval. **27**(2), 237–246 (2006)
16. Van Selm, M., Jankowski, N.W.: Conducting online surveys. Quality and Quantity **40**, 435–456 (2006)
17. Wing, J.M.: Computational thinking. Commun. ACM **49**(3), 33–35 (2006)

The Function of Note-Taking in Problem Solving in the Computer Science Escape Game Room-X

Alexander Hacke[1]([envelope]) and Nadine Dittert[2]

[1] Didaktik der Informatik, University of Potsdam, An der Bahn 2, 14476 Potsdam, Germany
ahacke@uni-potsdam.de

[2] Department für Informatik, Abt. Didaktik der Informatik, Carl von Ossietzky Universität, Uhlhornsweg 84, 26111 Oldenburg, Germany
nadine.dittert@uol.de

Abstract. The competence to solve problems is of fundamental importance in software engineering and the broader field of computer science. The escape game Room-X provides participating students aged between 13 and 19 years with an opportunity to tackle such a problem using computer science knowledge and general problem-solving skills. The problem-solving task presented in the game is representative of those encountered in professional computer science practice, and thus affords a valuable opportunity to examine the problem-solving processes of learners in this domain. This study focuses on the role of note-taking in the problem-solving process of the participants. Following an analysis of the relevant literature, we conduct a structuring content analysis of the notes taken during the game and examine how these notes are integrated into the problem-solving process. The findings suggest that note-taking is often employed as a simple memory aid and sees only limited use as a tool for organizing the problem-solving process. The study underscores the importance of developing effective note-taking strategies in computer science education to address the challenge of effectively organizing the problem-solving process, given the central role of problem solving in this domain. These findings are contextualized within the framework of computer science education and are discussed in relation to their broader implications for general education.

Keywords: computer science problem solving · escape game · note-taking · solution-construction tool

1 Introduction

Problem-solving skills are fundamental in computer science and must therefore be an essential element of computer science education in both schools and universities. While problem solving is recognized in the curricula of German schools,

the way in which it should be taught remains largely unspecified. The German educational standards for computer science address certain aspects of problem solving in various process and content areas, such as "Modeling and Implementing" and "Representing and Interpreting" and particularly in the content area of "Algorithms" [3]. In the state of Brandenburg's school curriculum, problem-solving primarily pertains to the domain of algorithmics, with the various factors influencing this process being outlined [12]. However, it remains unclear how these factors should be implemented in teaching, which ones are particularly important, and which measures can be taken in the classroom to ensure that students acquire the necessary competencies.

The reasons for the lack of specific teaching methods are manifold. Models such as Polya's [16] that attempt to illustrate problem-solving processes only partially coincide with reality because human problem solvers often do not proceed as systematically as the model suggests. They rather skip, overlap, and repeat individual phases. Moreover, such models can only be an attempt to reflect the complex thought processes involved but cannot provide an answer for all the variables involved. Schoenfeld has identified four essential prerequisites in this regard, namely knowledge, the use of heuristics, self-observation and regulation, as well as self-efficacy [17]. These manifest themselves to varying degrees depending on the problem and the individual, indicating that there is probably no universally applicable recipe for the acquisition of problem-solving competencies. However, at least the aspects that deal with knowledge of computer-science content as well as computer-science specific and more general problem-solving strategies and heuristics can be examined since these can indeed be part of the school education. This paper investigates a segment of these aspects, namely the question of the role of note-taking in the problem-solving process.

2   The Purpose of Note-Taking

Note-taking serves as a general strategy primarily aimed at capturing perceived information in written form. This information can be auditory, visual, or textual in nature. Note-taking is an integral component of classical learning environments and is indispensable in today's academic settings. The content, scope, and manner of note-taking have been extensively researched, particularly in the context of university lectures. At the latest since DiVesta's work in 1972 [4], two main functions of note-taking have been identified in this regard.

The first of these functions is that of an external storage. It serves to capture as much information as possible in order to have it available as a basis for learning later on and to be able to reconstruct any gaps in the notes during reading. In this type of note-taking, the focus is less on processing or organizing information, but rather on retaining as many facts as possible or preventing their forgetting. Several studies have shown that note-taking can contribute to better performance in examination situations (see Kiewra [8], for a review). In fact, the mere product of note-taking is already a conducive factor. Additionally, reviewing one's own notes can lead to even better performance in exam situations.

The second function of note-taking pertains to the active processing and organization of the written contents [13]. The written, yet not fully understood, information is linked to existing knowledge and imbued with meaning over time. Connections between different pieces of information must be identified, not-understood concepts must be paraphrased, or even reacquired from other sources and added to the notes in order to establish understanding. This process, referred to DiVesta [4] as encoding, ideally results in a representation of the writer's thoughts and mental models in relation to the information. The result is a better understanding and the ability to further process the information in one's own words. This process can be seen as an active learning process, where the material is linked with the writer's existing cognitive structure [4]. This second function of note-taking does not contradict the function of external storage, but rather builds on it. Hartley and Davies view this process and its resulting product as a form of analysis [7], which is located in the realm of higher-order thinking according to Bloom's taxonomy (cf. [2]). This taxonomy of learning objectives, in its revised edition by Krathwohl and Anderson, aims to categorize learning objectives based on the complexity of thinking involved. The three less complex categories are "remember", "understand", and "apply". The three more complex categories, "analyze", "evaluate", and "create", are often grouped together as higher-order thinking skills, indicating a greater level of cognitive demand expected from the learner. Therefore, it can be assumed that intensive processing of previously written material requires a certain level of cognitive maturity, but ideally provides assistance in learning. Note-taking is most helpful when it is done as completely as possible. Aiken showed that the part of the information that is not written down is unlikely to be recalled later [1]. However, according to Kiewra note-takers tend to omit large portions of the presented information, which impairs the encoding function of note-taking [8]. Two additional aspects that could influence the manner of note-taking in the future are the development of technological aids and the increasing ease of accessing information through the internet. Morehead et al. showed that at the time of their study, while much information was still being recorded on paper and with pen, notebook computers and similar devices had also become prevalent. In addition, the amount of note-taking is reduced, especially in online events and when presentation slides are available [13]. A survey of students conducted by Van der Meer et al. revealed that note-taking is often perceived as less necessary when information is readily available on the internet [11]. This indicates, on the one hand, that students are not always aware or familiar with the encoding function of note-taking. On the other hand, it suggests that note-taking may be utilized less frequently in the future due to the presence of easily accessible information.

3   Note-Taking During Problem Solving

A problem contains a multitude of variables that can play a role in finding a solution. The sheer number of variables involved exceeds the capacity of our

working memory, as noted by Muesseler et al. [14]. Moreover, the intermediate sub-results generated during the problem-solving process require side calculations. Therefore, it is sensible to make notes and jot down these sub-results. When compared to notes taken during lectures, notes taken during problem solving can initially be classified under the function of external memory storage. In this case, information is written down that is not yet understood and the relationships between the notes are unclear. Following the basic process of problem solving according to Polya71 [16], ideally, one begins with developing an understanding of the problem. All available variables should be considered in relation to the goal. Notes can be helpful in collecting all the details and generating an initial overview. Unlike lecture notes, there is usually no apparent structure at the beginning. The structure emerges only as the problem-solving process unfolds.

After the initial phase, the next step is to look for a solution approach. In this organizational phase [17], a plan needs to be developed from the available variables that leads to the solution of the problem. The previously taken notes can play a significant role in this process. They need to be interpreted, compared, and classified. If they are not recorded, details can be overlooked. The problem is broken down into smaller units, which must be related and assigned to the variables. Ideally, a mental structure of the problem emerges. This structure can be expressed in notes by mapping or clustering. The external memory that the originally recorded variables represented is now further processed in the encoding process. The usefulness of further processing lecture notes was demonstrated by King [9] and others. During problem solving, this phase of mental representation can also manifest itself in drawing diagrams or other graphic representations.

Problem-solving processes typically deviate from the idealized depiction proposed by Polya, wherein a sequential progression of problem understanding, plan formulation and execution, and solution verification occurs [16]. It is more likely that individuals may jump back and forth between phases, forget details, or misinterpret them, resulting in an incorrect mental representation of the problem. Nonetheless, notes are a good starting point for retracing initial thought processes or conducting error checks and reorganizing. Without notes, the risk of forgetting thoughts and intermediate results is significantly higher. Trafton and Trickett showed in a study that the use of notes during problem-solving and self-explanation led to increased learning and more accurate problem solving [18].

4   The Escape Game Room-X

4.1   Escape Games

Escape adventure games are immersive and challenging experiences in which teams of typically two to six individuals work together to collect clues and solve puzzles in order to unlock the door of a physical room in which they are confined. Often set in a story-driven scenario resembling a crime scene, players must locate and piece together various clues and objects to uncover the solution [15].

This engaging and motivating activity has already found its way into educational contexts, with escape games being used in various school subjects, including computer science (see [19]). Previous research has shown that problem-solving competencies can be addressed through this approach, and that learners' motivation and engagement in such a learning setting is generally high [5].

While educational escape games share many similarities with their recreational counterparts [15], there are notable differences in their aims and scope. In a comparison of different educational formats, approximately one quarter of escape games were found to focus on problem-solving skills as a learning objective [19]. To achieve this, it is necessary to embed the game within a broader problem that needs to be solved alongside other tasks.

4.2   Problem Solving and Note-Taking in the Escape Game Room-X

The escape game Room-X involves a problem of the "simple" category according to [14] with computer science-related content. This means that there is a clear starting and goal state, and some elements of the solution strategy involve tasks from the field of computer science. The game is situated within an educational context and primarily aims to investigate general problem-solving abilities. To ensure inclusivity and avoid excluding students who may lack specific computational problem-solving knowledge, the decision was made to refrain from incorporating problems that necessitate such expertise. Instead, the game includes subtasks that require a foundational understanding of computer science principles, thus preserving its inherent computational character. The objective of the game is to steal the next computer science exam from a fictional teacher and escape from an alarm-secured room within 60 min. Various computer science tasks must be solved to determine the password for a tablet containing the exam and a four-digit combination for a safe that holds the remote control to disable the alarm [6]. A secret message encrypted using the Caesar cipher must be decrypted into plain text, a Morse code needs to be deciphered using a binary tree, and errors need to be identified using the two-dimensional parity check method. Additionally, an automaton's acceptance of words must be determined based on its state diagram, and a bomb must be defused, which requires knowledge of binary-decimal conversion, understanding of source code, and propositional logic. Furthermore, the game incorporates notable figures from the field of computer science and highlights the complexity involved in solving a monkey puzzle as a form of combinatorial puzzle, thus extending the scope of computer science concepts explored within the game. None of these subtasks is accompanied by a specific written instruction. What needs to be done is derived from the examination of each respective object. Each team is composed of five or six participants who have been receiving computer science education for at least one year and are aged between 13 and 19 years. To facilitate note-taking by students, the room is equipped with two whiteboards. Following the prevailing practice in the surrounding schools, one of the whiteboards is of the conventional type, while the other is a digital board. This arrangement serves not only

to enhance the immersive nature of the classroom setting but also allows students to utilize their preferred note-taking system.

In order to solve the problem, participants must find various objects, solve computer science-related tasks, and construct a path to the solution based on the information obtained. Since the purpose of the objects and partial solutions may not be immediately obvious, note-taking is a useful tool, particularly since the game is played in teams, and this allows all participants to access the information gathered. Noteworthy is that the activity of note-taking should be intertwined with the problem-solving process as a continuous activity and not limited to specific moments or tasks.

After this, hypotheses should be formulated regarding how the information can be distinguished from one another, how it relates to each other, and which of the sub-goals they may be relevant to. The resulting content on the board ideally corresponds to the path through the problem space. This second phase in the problem-solving process according to [17] can thus be supported by encoding the notes.

5   Research Methodology

Previous research has shown that note-taking plays a significant role not only in learning but also in problem-solving processes. Depending on how they are used, notes can contribute to a deeper understanding and ultimately to success in solving a task.

Therefore, in the field of computer science, the question arises as to the role that note-taking plays in students' problem-solving processes. This will be investigated using the aforementioned computer science escape game, where the following two sub-questions are intended to contribute to clarification:

1. What types of notes do participants take during the escape game?
2. How do participants integrate their notes into the problem-solving process during the escape game?

To address these research questions, the notes taken by participants during the escape game Room-X, who agreed to audio and video recording of their activity, were analyzed. The data analyzed here consist of notes made by participants on the whiteboard and the electronic board while they were solving the tasks. For analysis purposes, photos and screenshots of the notes were used, which varied in number per run depending on the changes made by the participants. In some groups, only one photo of the whiteboard was used for analysis, while in others, multiple photos or screenshots were used. The decision on the number of photos or screenshots depended on the extent to which individual notes were erased, replaced, or modified. A total of 54 runs were considered. Four of the 54 data sets used were removed because they either contained no notes or no analyzable notes. The data were collected between May 2018 and January 2023, with no data available between March 2020 and October 2022

due to pandemic-related restrictions. During this time period, the game and the tasks remained unchanged.

The analysis of the data was carried out by two researchers using a structuring qualitative content analysis based on Mayring's approach [10]. Previous research has shown that notes can either serve as external memory, simply acting as an extension of one's memory, or as material for encoding, reflecting, and transforming information, serving as tools for active engagement with the content. To approach the question of the role of notes in problem solving in the context of the escape game, a deductive category application approach was used to examine whether these structures were present in the analyzed data. The structuring dimensions derived from the literature and found in the data are henceforth referred to as *storage* and *working tool*, respectively. They are intended to contribute to the answer to the first research question. To gain a more detailed understanding of how participants use their notes, we searched for specific manifestations of both dimensions and established a category system based on these findings, which serves to answer the second research question. Hartley and Davies categorize note-taking as an example of analytical thinking when they identify activities such as (1) identifying elements, (2) identifying relationships between elements, and (3) identifying an organizational structure of the material [7]. Translating these categories to note-taking in the escape game, we define them as follows: identifying elements refers not only to the act of noting down found information, but also to evaluating and working with the information to develop clues (Category 1). In the escape game, players must initially locate and identify information as useful or not useful, which requires active engagement with the notes to develop clues. This includes results from (computer science) tasks as well as recognizable solution paths, such as deciphering the Caesar ciphered secret text. Elements that do not contribute to progress in solving tasks must also be identified as such, which can be observed in the notes through erasing or striking out irrelevant information. The active identification of elements suggests that the board is used as a tool, while the mere recording of found information without any editing or deletion indicates note-taking as a storage mechanism. Relationships between elements (category 2) are relevant for solving tasks and were also found in the data. This includes relationships between elements or to the goal of the activity. Notes falling under the category of "element relationships" include cases where there is a clear connection between different notes, such as 14 dashes serving as a placeholder for a password with a reference to additional clues stating "11 letters, 2 special characters, 1 number." When a relationship to other notes or references to them is visible and the notes help to draw further conclusions or serve as a basis for further work, this suggests that the board was used as a working tool. Notes without a relationship to others or with only a connection to their source (e.g., "Floor → Key", meaning the key was found on the floor) do not indicate analytical thinking and rather suggest the use of notes as storage.

The organizational structure of the material (Category 3) can also contribute to answering the question of the integration of notes in the problem-solving pro-

cess. As the escape game Room-X consists of two main tasks, one would expect a visible division of the notes into these two areas. If notes are placed according to whether they contribute to the solution of the password for the tablet or for opening the safe, a structure becomes apparent, indicating a systematic approach. Also, placing elements close to each other that belong to the same (partial) tasks reveals a structure. If, on the other hand, notes are placed freely and without any recognizable connection, the structure is lacking, which suggests that the boards are used more as storage for information rather than as a working tool for problem solving, as in the case of the structured arrangement.

6   Findings

The analysis of the notes taken during the escape game revealed the well-established dichotomy between their use as a working-tool for problem solving or as storage. Out of the 50 data sets examined, 30 were classified as storage, 19 as a working tool, and one data set was disputed between the researchers. Figure 1 clearly illustrates how the boards were used as working tools for problem solving. It can be observed that elements were identified and worked on in a structured manner with visible connections. The development of specific content is visible on the electronic board: the encrypted text and its corresponding plain text "ABSTEIGEnd" ("descending") were noted, as well as the Morse code and its corresponding decoding. The structure is clearly evident, with everything related to the password noted on the electronic board (top row) and all notes related to the door code on the whiteboard (bottom row). Hints that were initially placed in the wrong location were later erased and transferred to the correct place. For example, "absteigend" ("descending") was recognized as belonging to the door code and was transferred to the board where the information regarding the door code was collected and erased from the other board. Conversely, "Bill 28" was transferred from the whiteboard to the electronic board and then removed from the original location. Additionally, the participants established connections by drawing lines between hints, such as linking "Buchstaben 6 + 5 Fehlen" ("letters 6 + 5 missing") to "14 Zeichen" ("14 characters"). The progress of work was also documented, as evidenced by the strikeout and subsequent removal of "Schlüssel auf Boden" ("key on floor"). Likewise, a question mark behind "1 Ziffer" ("1 digit") was removed after it was found.

Figure 2 represents an example in which the electronic board was used solely as a storage device for information. The analysis of these notes reveals that the progress of work was not documented, and that the notes were unstructured and lacked any reference to further actions. The word "ABSTEIGEND" ("descending") needs to be decrypted using the Caesar cipher in the escape game, which apparently happened, but was not recorded in the notes. Furthermore, it remains unclear what purpose this hint serves, as no relation to it was noted. Other notes, such as "Endzustand Doppelkreis" ("finate state double circle") or "14 = Zeichen, 2 = Sonderzeichen, 1 = Ziffer" ("14 = characters, 2 = special characters, 1 = digit"), were merely copied from found clues or work instructions. Again,
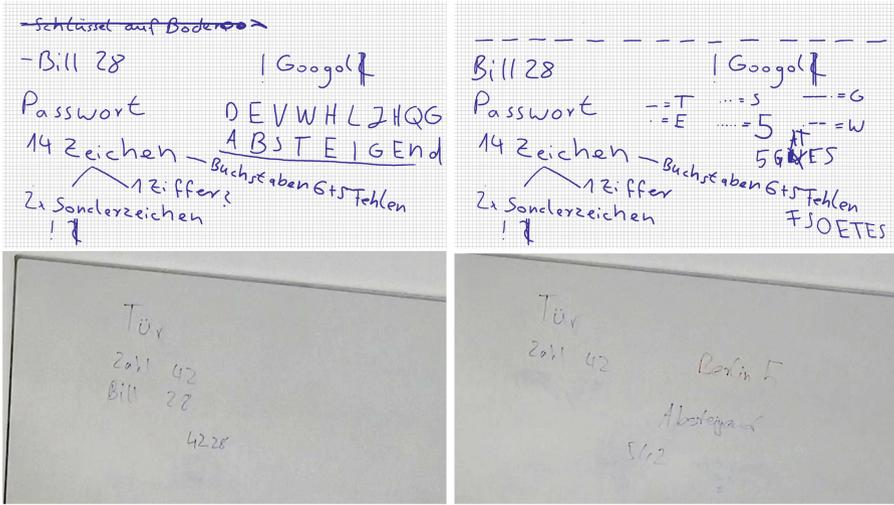
Fig. 1. Exemplary presentation of notes as working tool at different points in time during the game. Top left to bottom right: Minute 35, 59, 28 and 33 out of 60.

there is no apparent connection to what these notes are for. The only reference that was noted is to the origin of a part of the task "Boden → Schlüssel" ("Floor → Key"), which, however, has no relevance for the further solution process. The notes in this example lack any structure that would indicate which clues belong together or lead to the solution of which sub-puzzles. Additionally, the example contains partial results that are of little use in their current form. The underlined item "5 ... GATES" contains a part of the password for the tablet, but it is incorrect and the entire password remains incomplete.

The use of a category system enables a clear classification of notes as either working tool or storage. The calculation of interrater reliability using Cohen's Kappa yielded a value of 0.96. Out of the 50 data sets considered, 30 were categorized as storage and 19 as working tool. There was a disagreement between the two researchers regarding the categorization of one data set. It is worth noting that the datasets categorized as working tool also encompassed the function of storage. However, not all notes are as comprehensive as the examples presented. Even in less organized examples, helpful approaches for solving puzzles can become apparent. For example, in some cases, the information about the structure of the password was visualized by drawing 14 strokes on the board, which corresponded to the combined length of the two password parts. However, the two password parts were not linked to each other afterwards. The same applies to the word "ABSTEIGEND" ("descending"), which is frequently found on the boards, even though there might not be any other notes. These examples demonstrate that the types of notes are not always sharply distinguishable from each other. The available data does not provide a clear demonstration of a direct connection between the type of notes taken and the success in the escape game.
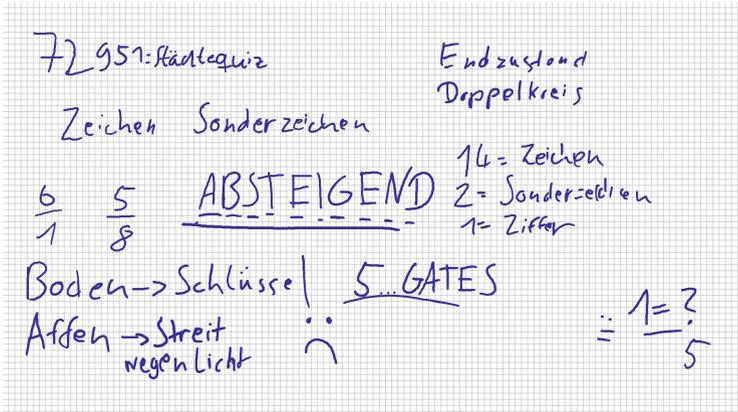
Fig. 2. Example of note-taking as a storage; screenshot taken at minute 59 out of 60 of the game.

7 Discussion

In summary, it can be concluded that in a problem-solving situation, as well as in note-taking situations in lectures, the distinction between external storage and encoded material becomes apparent. This suggests that note-taking serves not only a similar purpose but also a similar benefit, namely that of deeper understanding. However, it should be noted that lecture notes already have a predetermined structure, while in the escape game, the structure needs to be created on a completely blank sheet. Hence, in this case the activities to process and to organize the information must be accomplished by the learners themselves. Although the categorization system from lectures was found here as well, the types of notes are not always clearly distinguishable which indicates a potential transition from storage to working tool. This suggests that some participants who use notes for external storage can also use them in a structured manner, albeit to a limited extent. Nevertheless, our study indicates that the encoding phase often remains in its early stages, raising the question of why this process stalls. The game environment imposes certain constraints that influence the process of note-taking. In a problem-solving scenario where time pressure is a factor, it may seem more intuitive to start immediately rather than dedicating time to note-taking. Additionally, the situation demands heightened cognitive efforts, leading participants to seek ways to mitigate the cognitive load. However, effective note-taking can actually aid in reducing the cognitive burden by alleviating the need to retain all variables mentally, ultimately resulting in a net time gain, as it enables faster resolution of confusion. The question arises how computer science education can contribute here as this concerns problem solving competencies. A deeper look into computer science education contents offers some aspects that could be addressed. Firstly, a lack of competencies are evident in the process area of "Representing and Interpreting" [3], as indicated by the

relatively underdeveloped representations. Deficits are also visible in the process area of "Structuring and Networking" [3], particularly in the achievement level II "Reorganization and Transfer" of this process area, which manifests in the lack of systematic approach as well as the less developed structures in the representation of the content. This particular aspect is a fundamental skill. When confronted with a problem, it is crucial to analyze the structure of the initial situation in a deliberate manner, aiming to gain a comprehensive understanding and develop a structured approach towards problem solving. However, the nature of the notes from our study suggests that considerations of the overall structure typically emerge late in the problem-solving process and tend to remain focused on searching or individual tasks. Engaging in visualization and actively working with the notes can help shift the focus towards the broader structural aspects. In the "reflection and problem-solving" achievement level of the process area, the analysis of the notes also revealed that only a few of the students are able to visibly structure their knowledge and their approach to problem solving.

For computer science education, our results can contribute as a starting point for developing these competencies, through which students can be given a universal tool for problem analysis. By implementing note-taking as a working tool, problem-solving processes can be approached more easily, thereby reducing the initial hurdle. Creating connections between different items of information and linking them to the writer's cognitive structures might be a part of the learning process that should be the focus of future research. One approach for computer science education could be to expose students more frequently to unfamiliar problems and refrain from providing specific solution steps. Instead, the focus could be on practicing systematic problem analysis and decomposition, utilizing visualization techniques and structured note-taking. This approach aims to facilitate the initial transition from a daunting unknown to the development of a concrete computational approach.

It should be noted, however, that the use of notes alone was not decisive for the success of the teams in the escape game. Various other factors, such as team composition, communication within the team, prior knowledge of computer science concepts, motivation, and self-regulation, also played a significant role. Nevertheless, a correlation between the use of notes and progress in problem solving appears likely, and the results can be interpreted in line with those of Trafton and Trickett [18], whereas the use of a free-form note-taking system was helpful depended on whether users could integrate it into the problem-solving process.

If the goal is to promote the use of note-taking in computer science education with regard to problem solving, it is first necessary to understand the reasons for non-use or misuse and then investigate how note-taking can be actively promoted in the problem-solving process. It is reasonable to assume that acquiring this competence cannot be the sole responsibility of computer science education, but rather cross-disciplinary solutions are desirable.

References

1. Aiken, E.G., Thomas, G.S., Shennum, W.A.: Memory for a lecture: effects of notes, lecture rate, and informational density. J. Educ. Psychol. **67**, 439–444 (1975)
2. Anderson, L.W., Krathwohl, D.R.: A Taxonomy for Learning, Teaching, and Assessing: A Revision of Blooms Taxonomy of Educational Objectives. Longman (2001)
3. Arbeitskreis Bildungsstandards SII: Bildungsstandards Informatik für die Sekundarstufe II. supplement to LOG IN 183/184 (2016)
4. Di Vesta, F.J., Gray, G.S.: Listening and note taking. J. Educ. Psychol. **63**(1), 8–14 (1972)
5. Fotaris, P., Mastoras, T.: Escape rooms for learning: a systematic review. In: Proceedings of the European Conference on Games Based Learning, pp. 235–243 (2019)
6. Hacke, A., Przybylla, M., Schwill, A.: Beobachtungen zum informatischen problemlösen im escape-adventure-spiel "room-x". Informatik für alle (2019)
7. Hartley, J., Davies, I.K.: Note-taking: a critical review. Programmed Learn. Educ. Technol. **15**(3), 207–224 (1978)
8. Kiewra, K.A.: Investigating notetaking and review: a depth of processing alternative. Educ. Psychol. **20**(1), 23–32 (1985)
9. King, A.: Comparison of self-questioning, summarizing, and notetaking-review as strategies for learning from lectures. Am. Educ. Res. J. **29**(2), 303–323 (1992)
10. Mayring, P.: Qualitative Inhaltsanalyse : Grundlagen und Techniken. Beltz Pädagogik, 12, vollständig überarbeitete und aktualisierte aufl. edn. (2015)
11. van der Meer, J.: Students' note-taking challenges in the twenty-first century: considerations for teachers and academic staff developers. Teach. High. Educ. **17**(1), 13–23 (2012)
12. Ministerium für Bildung, Jugend und Sport des Landes Brandenburg: Rahmenlehrplan f. d. unterricht i. d. gymnas. oberstufe i. l. brandenburg (2018)
13. Morehead, K., Dunlosky, J., Rawson, K.A., Blasiman, R., Hollis, R.B.: Note-taking habits of 21st century college students: implications for student learning, memory, and achievement. Memory **27**(6), 807–819 (2019)
14. Müsseler, J., Rieger, M.: Allgemeine Psychologie, 3rd edn. Springer, Berlin (2017)
15. Nicholson, S.: Peeking behind the locked door: A survey of escape room facilities (2015)
16. Polya, G.: How to Solve It. Princeton University Press, Princeton (1971)
17. Schoenfeld, A.H.: Reflections on problem solving theory and practice. Math. Enthusiast **10**(1), 9–34 (2013)
18. Trafton, J.G., Trickett, S.B.: Note-taking for self-explanation and problem solving. Hum.-Comput. Interact. **16**(1), 1–38 (2001)
19. Veldkamp, A., van de Grint, L., Knippels, M.C.P., van Joolingen, W.R.: Escape education: a systematic review on escape rooms in education. Educ. Res. Rev. **31**, 100364 (2020)

An Exploratory Investigation on High-School Students' Understanding of Threads

Emanuele Scapin[1]([✉]) [iD], Nicola Dalla Pozza[1] [iD], and Claudio Mirolo[2,3] [iD]

[1] ITT G.Chilesotti, 36016 Thiene, Italy
{escapin,ndallapozza}@chilesotti.it
[2] University of Udine, 33100 Udine, Italy
claudio.mirolo@uniud.it
[3] Lab. CINI "Informatica e Scuola", Rome, Italy

Abstract. Students' difficulties to learn concurrent programming are well known amongst Computer Science instructors. While in the International Computing Education community it is still up to debate the extent to which such topic should be included in pre-university curricula, based on our country's Ministerial guidelines for technical high schools with a specialization in Computer Science, students are expected to acquire key concurrent programming skills. With the aim of getting insights about the nature of students' difficulties, as well as to identify possible pedagogical approaches to be adopted by teachers, we have undertaken an investigation on students' perception, proficiency and self-confidence when dealing with concurrency and synchronization tasks. We then present the results of a preliminary study carried out by submitting a survey in a couple of representative high schools of our area. The survey includes subjective perception questions as well as small program comprehension tasks addressing students' understanding of thread synchronization. Moreover, we also analyze students' self-confidence in connection with their actual performance in such tasks. A total of 68 high school students were engaged in the survey. Our findings indicate that students' perception of self-confidence tends to weakly correlate to their actual performance, although more in general they express a low self-confidence level in relation to the topic. In particular, the results clearly show that the concept of thread synchronization is especially difficult to master for a large majority of them.

Keywords: Informatics education · Programming learning · High school · Threads · Concurrent programming

1 Introduction

Over the last decades, in order to achieve ever increasingly powerful architectures, the processor industry has shifted its manufactures towards multi–core

and many–core designs. This trend has become so widely adopted that nowa-days such hardware is employed not only for industrial and academic purposes [8], but it is also readily available in desktop machines in school laboratories [2]. Alongside to the electronic improvements, to harness the computational power of these machines it is necessary to design software solutions that exploit the numerous cores with concurrent programs.

There is therefore the need for experienced programmers leveraging this paradigm, as indicated, in particular, by the ACM/IEEE Task Force in the Computer Science Curricula 2013 Report [12]. Academic institutions often introduce concurrent programming in advanced dedicated courses or in conjunction with related topics, e.g. [6,11,21], but less commonly in introductory programming modules [3,13]. As a matter of fact, it is still up to debate when and how to approach the subject, and even more open to debate is the feasibility to cover concurrent programming in the high school curriculum [1,2,14,23]. Students' difficulties to learn concurrent programming are indeed well known among both high school teachers and university instructors, and are also confirmed by a handful of empirical investigations, such as [2,10,14].

According to the Italian Ministerial guidelines, the students opting for a technical specialization in Computer Science are expected to develop parallel and concurrent programming skills by the end of their fourth high school year (grade 12).[1] Unfortunately, however, there is a lack of empirical evidence to set realistic curricular goals. As pointed out by Brođanac et al. in a recent paper, "[i]n the case of teaching parallel programming before university level, the research appears to be scarce" [2, p. 2].

Within the outlined framework, this work is meant as an initial, exploratory investigation on students' proficiency and self-confidence when dealing with concurrency and synchronization tasks. More specifically, we are trying to find at least some preliminary answers to the following research questions:

Q1. To what extent are students at ease with some basic concepts of concurrent programming?
Q2. To what extent does students' perception of self-confidence correlate with their actual performance in simple concurrent programming tasks?
Q3. What are their major difficulties when learning concurrent programming?

In order to address the above questions, we have administered a survey including subjective perception questions as well as four program comprehension tasks involving basic aspects of thread synchronization. A total of 68 students attending the last year (grade 13) in representative high schools of our area were engaged in the survey. The students were introduced to concurrent programming in the previous school year, by different teachers who may have used diversified approaches.

The paper is organized as follows. In Sect. 2 we summarize the background of this work. Section 3 is about the survey structure and the rationale underlying its design. The main findings of our exploratory investigation are then outlined

---

[1] https://www.gazzettaufficiale.it/eli/gu/2012/03/30/76/so/60/sg/pdf.

in Sect. 4 and discussed in Sect. 5. Finally, in Sect. 6 we draw some conclusions and mention possible future developments of the present work.

2   Background

At least anecdotally, students' difficulties with concurrent programming are acknowledged by technical high school Computer Science teachers, as well as by lecturer at the undergraduate and graduate levels. Nevertheless, this subject has hardly been given significant attention in the context of introductory programming. For instance, in their systematic study [17] reviewing over 700 research papers published between 2003 and 2018, Luxton-Reilly and colleagues mention only two contributions addressing parallel or concurrent programming.

In the following two subsections we will mention a selection of works specifically focused on the topic, respectively at the tertiary and secondary instruction levels, whereas for a broader literature analysis the reader is referred to the recent review discussed in Brođanac et al.'s paper [2]. We will then conclude this background section with a couple of notes about the Ministerial guidelines that apply to our school system.

2.1   Parallel and Concurrent Programming at the University Levels

A range of works on the learning of concurrent programming at the undergraduate or graduate levels examine students' performance, their common difficulties, their understanding of and misconceptions on the subject. For instance, Choi and Lewis [5] analyze and classify the pitfalls in a collection of simple multi-threaded programs written by students in order to improve instruction and develop learning aids. While focusing on thread-safe Java classes, Fekete [7] identifies suitable learning outcomes and discusses related pedagogical difficulties, also proposing examples that could help students avoid common misconceptions. Lönnberg & Berglund [16] investigate the defects of concurrent programs produced by students from a program development perspective.

A recurrent theme at the undergraduate level is whether these topics should be taught in a dedicated course [16,21], split into multiple units [11,13], or covered in an introductory programming course [3,13]. In this respect, Zhu et al. [25] argue that the conceptual shift from sequential programming to concurrent and parallel programming is notoriously difficult to make. The authors present their results of using the educational game Parallel, focusing on the learners' self-efficacy and how they learn concurrency concepts, and show that undergraduates' self-efficacy correlates with the time students spend in multi-threaded problem-solving. Formerly, Bruce et al. [3] had suggested making use of graphics and animations in order to facilitate student learning through visual feedback.

No matter how challenging the subject is, Gardner argues that, in light of the current developments of multi-core architectures, refraining "from teaching parallel programming to CS undergraduates is a kind of educational malpractice,"

since computer technology cannot be expected "to turn back to the old, comfortable path of ever-increasing uniprocessor clock speeds [...]. To be prepared for careers in this emerging environment, our students need to be furnished with the knowledge and practice of parallel programming" [9, p. 3:6]. And Rivoire [21] reported that upper-level undergraduates were indeed interested in and satisfied with the contents of an introductory course on multi-core programming models. However, according to Ko et al.'s teaching experiences in multiple courses [13], 1st and 2nd year students can recognize parallelism in programming tasks and are generally aware of synchronization issues (although would prefer to approach this subject in later years). In addition, Conte et al. [6] claim that the achievements in parallel programming of novices without previous exposure to computing concepts appear to be comparable to those of more advanced students.

2.2   Parallel and Concurrent Programming in High School

Although concurrent programming has usually been considered exceedingly challenging in a pre-tertiary context, sporadic attempts to introduce this topic in high school date back to the mid '90s. According to Rifkin, for instance, "it is never too early to teach so-called 'hard' concepts" such as basic principles of parallel algorithms and software engineering, providing the ideas are presented "in a manner that is simple, fun and suited to the audience" [20, p. 26].

Much work on the teaching and learning of concurrent programming in the upper secondary school has been done by the Israeli CS Education community. In particular, in the context of a high school unit in concurrent and distributed computing, Ben-Ari & Kolikant [1] explored the evolution of students' conceptions and attitudes, and found that they were eventually able to develop parallel algorithms and to prove their correctness. Although the involved students initially felt extremely challenged, they then came to appreciate the relevance of the topic and its contribution to improving their cognitive skills. Later, Kolikant observed that, although high school students are able to gain a "rich" understanding of various synchronization tasks, quite often their successful solutions to synchronization problems are achieved by a pattern-based approach "exempting them from dealing with the dynamics of the synchronization mechanisms," so that the underlying "concepts become inert" [14, p. 243].

Also in Tobert et al.'s view [23] there is ample evidence that parallel algorithmic thinking and multi-threading can be taught—and should be broadly covered in CS education—as early as high school. Moreover, Brođanac et al. [2] have recently conducted an investigation in 3 Croatian high schools where parallel programming was included in the informatics curriculum. They report getting positive feedback from students as to the interest and usefulness of the learnt content, even though it was perceived as more difficult compared to the other topics. The authors conclude that parallel programming, including synchronization issues, can be taught in a high school context at least as an optional subject.

2.3   High School Ministerial Guidelines in Italy

We conclude the background section by briefly summarizing the implications of the Ministerial guidelines framing our Country's secondary school system. Rather than prescribing a detailed syllabus, such recommendations are meant to be a reference for the teaching of each covered subject. In the case of technical high schools with specialization in Computer Science, the guidelines are articulated in terms of *knowledge*, *skills* and *competencies* to be achieved in the second biennium (grades 11 and 12) and in the last year (grade 13).

---

1. General information (2 questions)
gender; subjective preference of programming languages for use with threads

2. Thread-related concepts in general (7 subjective perception questions)
difficulty of the subject in general; self-efficacy in general; adequacy of time spent on the subject; adequacy of program examples; difficulty with specific thread-related concepts/issues; difficulty with specific thread-related (technical) program tools; difficulty with thread synchronization

3. Tasks (7 program comprehension questions and 4 evaluations of self-confidence on the provided answers)
Task1 – mutual exclusion and race conditions (4 related questions); Task2 – producer-consumer scheme; Task3 – synchronization analysis; Task4 – deadlock analysis

4. Possible learning aids (3 subjective perception questions)
whether graphical representations have ever been considered; expected potential of graphical representations; perception of the use of specific graphical representations

5. Open suggestions (1 question)
suggestions to make the lessons on threads more interesting and clearer

---

Fig. 1. Structure of the survey. More details in Appendix: Survey Questions

As to the second biennium, in terms of knowledge students are expected to learn the components of operating systems, including techniques and technologies for concurrent programming and for the synchronized access to shared resources; in terms of skills, they should eventually be able to design and develop applications that interact with the operating system, using whenever appropriate concurrent programming strategies. In their last high school year, students should acquire knowledge on methods and technologies for network programming, as well as on protocols and communication languages of the application layer; in terms of skills, they are trained to develop applications that leverage network communication, such as client-server applications through simple communications protocols.

Within this framework, while teachers still have the freedom to personalize the course organization, concurrent programming is nonetheless considered an essential part of the high school curricula in Computer Science.

3   Instrument

In this section we outline the survey structure and the rationale underlying its design. The general organization, summarized in Fig. 1, is partly drawn from a similar instrument developed and used by the authors to get insight about the learning of other programming concepts [22]. Overall, it includes 24 items, 11 of which are based on 4 small tasks—7 program comprehension questions and 4 evaluations of self-confidence on the provided answers in a 4-grade Likert scale. The full survey is reported in the Appendix: Survey Questions.

The questions addressing the learning of concurrency concepts elaborate on Choi and Lewis' [5] "catalog" of errors that students typically make when approaching multi-thread programs, particularly in connection with data race conditions, deadlocks and other synchronization issues. To accommodate for the common practice in the involved high schools, the code is Java-like. The first two tasks draw inspiration from programs proposed by Meyer at al. in *"Concurrent Programming with Java Threads"*,[2] and are intended to test students' ability to anticipate the outcomes of concurrent threads. The other two tasks elaborate on Fekete's work [7]: the third task is meant to see if students are able to identify, among 5 options, both (equivalent) appropriate code fragments to deal with a shared resource; the fourth one requires to recognize deadlock-prone code and figure out suitable corrections.

For the solutions of each of the four tasks, the students were also asked to indicate their perceived level of self-confidence in a Likert scale ranging from 1 (not confident at all) to 4 (fully confident). We included these items because of the potential pedagogical implications pointed out, e.g., in [18,19], as well as in order to assess the extent to which students' subjective perception of difficulty correlates with their actual performance on a task. The concluding survey sections concern potential graphical/visual tools that could help understand concurrent programming and possible additional suggestions to improve the instructional practice. A particular interest in graphical/visual tools is motivated by recent research findings suggesting that students of STEM domains are more likely to exhibit a visual-spatial cognitive style [24].

4   Results

Data Collection

The survey was administered to 68 fifth-year students (grade 13: age range 18–19; 60 boys and 8 girls) attending a CS curriculum in two technical high schools in the North-East of Italy. The students, who were taught concurrent programming in the previous school year, engaged in the task in a controlled situation, under the supervision of their teachers. They were expected to complete the survey

---

[2] Course material available at the link https://se.inf.ethz.ch/courses/2011a_spring/soft_arch/lectures/old/13_softarch_self_study_threads.pdf.

within an hour. The answers to the survey were registered only anonymously, and then could not be used for formative or summative assessment.[3]

General Thread-Related Concepts

Here we summarize the results of survey Sect. 2 (see Fig. 1). To begin with, more than two thirds of the students find thread programming difficult (57%) or very difficult (12%) and report to be unsatisfied (54%) or completely unsatisfied (13%) with their performance in thread-related tasks. Conversely, most of them consider the time spent to deal with the subject (67%) as well as the proposed programming tasks (63%) to be both adequate to the purpose.

At a finer level of granularity, the bar chart in Fig. 2 shows students' perceived difficulty for diverse thread-related concepts.   As can be seen, *synchronization*
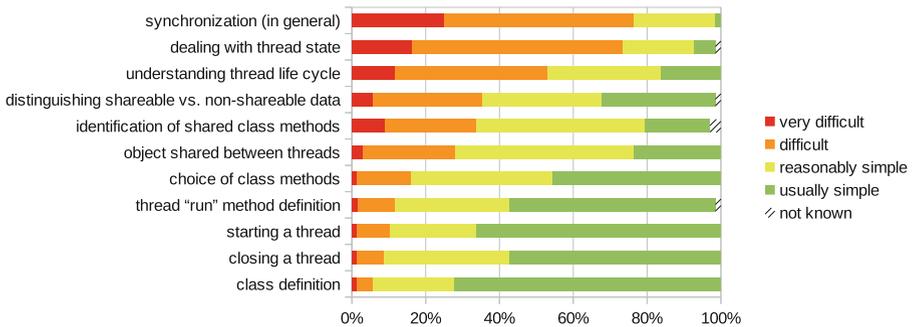
Fig. 2. Students' difficulties with a range of thread-related concepts.

and dealing with *thread states* are regarded as difficult to learn by 77% and 74% of them, respectively, whereas several aspects in connection with the organization of the classes implementing threads seem less critical. Although to a lesser degree, the answers to the following question, using a similar Likert scale and concerning a range of specific technical program tools, essentially confirm this picture: looking at the hardest side, the Java keyword `synchronized` turns out to be difficult for about 40% students and the methods `wait` and `notify`/`notifyAll`, governing the thread state, for roughly one third of them. The last question of survey Sect. 2 focuses on managing shared resources. All such related aspects are perceived as difficult by a significant percentage of students, ranging from about 30% (reading/writing operations) to about 60% (synchronization operations).

---

[3] In particular, since no personal information was shared with any third party, the Italian research policies do not require the approval by an ethics commission.

Program Comprehension Tasks

In summary, the first task presented a very simple class aimed at synchronizing the access to a shared resource, based on availability of data. Four temporal sequences of methods invocations by two concurrent threads were then presented, with a request to identify the resulting outcomes (questions a–d). The second, more complex task was about an instance of the *producer-consumer* scheme; again, students were asked to identify the correct output. In the third task, the (two) sound implementations of a straightforward synchronization scheme were to be recognized among five options. Finally, the last problem asked to choose a suitable strategy, informally described in words, to fix a given deadlock-prone code. (The full text of these tasks can be found in the appendix.)

In Fig. 3a are reported the percentages of (fully) correct answers to the seven questions asked for tasks 1–4. As we can see, only the solution of subtasks 1a and 1b are correct for a large majority of students; in all the other cases, less than half of them was successful, with the worst performances taking place for the apparently easy subtasks 1c and 1d. Figure 3b depicts the overall distribution of self-confidence levels between correct and incorrect answers. In this case, since there was one such Likert evaluation for each task, the solution of task 1 is considered correct when all four related answers are correct. Besides evidencing that less than one third of the students are more confident than not about their answers, what once again confirms their difficulties with concurrent programming, the diagram clearly shows that students' self-confidence in the provided solutions is only weakly connected with their actual achievements.


(a)

(b)
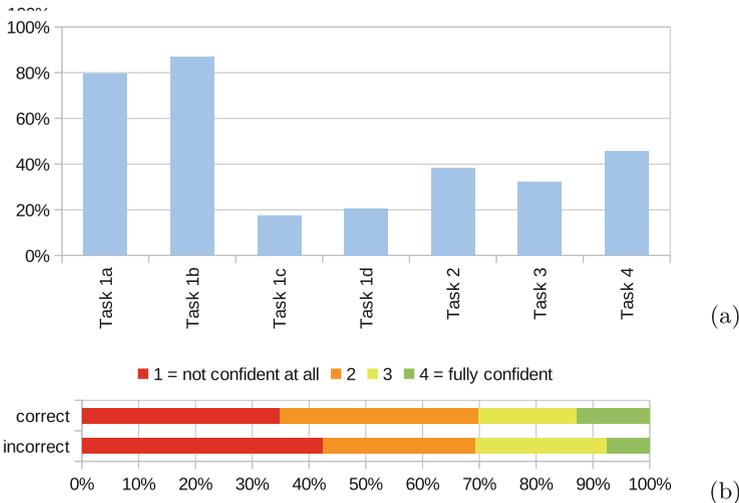
Fig. 3. Students' performance in the proposed tasks and perceived self-confidence in connection with their correct and incorrect answers.

We also tried to look in more depth at the relationships between quality of students' answers and perceived levels of self-confidence. To this aim, we scored the performance on task 1 in terms of number of correct solutions to questions a–d, and the performances on the other tasks by distinguishing three quality levels: 1 (= severely incorrect), 2 (= incorrect) and 4 (= correct) for tasks 2 and 4; 1 (= incorrect), 3 (= partly correct) and 4 (= fully correct) for task 3. The correlations and the corresponding statistical significance are summarized in Table 1, which should be self-explanatory.

Table 1. Correlation between students' performance in the tasks and their perception of self-confidence in the provided answers (in a Likert scale 1–4); the correlation cannot be taken as statistically significant if p-value > 0.05.

Task	correlation	p-value
Task 1 (number of correct answers to questions a–d)	0.278	0.0215
Task 2 (4 = correct, 2 = incorrect, 1 = severely incorrect)	0.440	< 0.0002
Task 3 (4 = correct, 3 = partly correct, 1 = incorrect)	0.150	0.2234
Task 4 (4 = correct, 2 = incorrect, 1 = severely incorrect)	0.017	0.8917
Task 1–4 (average score vs. average self-confidence level)	0.365	0.0022

Possible Learning Aids and Additional Suggestions

The last section of multiple-choice questions focused on possible graphical/visual learning aids. 59% students reported having thought about using graphical diagrams, and as many as 88% believe that a graphical representation could be effective to improve their understanding of concurrent programming. However, fewer students have a clear idea about which type of tool would be best suited to deal with thread-related concepts. After all, the most effective tools that may perhaps be used are not widely known in school contexts: in particular, several students did not know about Petri nets (74%), Wait-for/Holt graphs (51%), or finite-state automata (50%). Thus, the tools considered most useful for understanding were flow-charts (51% of positive ratings) and block diagrams (43%), even though they are not the best suited to the purpose.

The final open answer was answered by 45 students and 12 of them suggested to introduce examples of increasing difficulty more gradually (e.g. *"exercises and examples of more gradual difficulty"*). Other recurrent proposals include the use of concrete, real-world examples (6 – e.g. *"examples and exercises drawn from the real world [...]"*); the use of graphical/animation aids (6 – e.g. *"make more use of graphical representations [...]"*); a deeper theoretical discussion.

5   Discussion

First of all, we discuss the research questions raised in the introduction.

Q1 – To what extent are students at ease with some basic concepts of concurrent programming? All findings summarized in the previous section—answers to subjective perception questions, actual performance in the proposed tasks and self-confidence in the provided solutions—consistently indicate that concurrent programming is a rather challenging subject for high school students. This is not surprising, in that it corroborates what other educators have observed, by analysing both learners' performance (more often at the tertiary instruction level, e.g. [5,23]) as well as their subjective perception. In the latter respect, in particular, Brođanac and colleagues [2] report that concurrent programming is perceived by the high school students involved in their investigation as more difficult than several other programming topics.

Q2 – To what extent does students' perception of self-confidence correlate with their actual performance in simple concurrent programming tasks? To the best of our knowledge, this kind of analysis does not appear in previous studies specifically addressed to concurrency. The statistics listed in Table 1 indicate that, overall, students' self-confidence in the provided solutions tends to only roughly correlate to their actual performance in the tasks at hand—and, more in general, they express a low self-confidence level in relation to the considered subject. The discrepancy between self-confidence and performance is especially marked relative to the first task, where higher levels of self-confidence align with wrong answers for subtasks 1c and 1d—so suggesting some lack of awareness about their difficulties, even for a simple problem.

Q3 – What are their major difficulties when learning concurrent programming? As pointed out in Sect. 4, dealing with *synchronization* and with thread state transitions (via *wait/notify* operations) represent major challenges. Such difficulties emerge both from students' perception and from their performance in synchronization tasks. Once again, this confirms previous results that synchronization mechanisms are common sources of students' mistakes, see e.g. [5,7]. In connection with synchronization tasks, it may also be worth observing that students' performance in subtasks 1c and 1d is significantly worse than that in task 2 (see Fig. 3), although the program in the latter case is far more complex. A conceivable explanation of a similar phenomenon is that envisaged by Kolikant [14] and mentioned in Sect. 2: a successful solution for task 2 may be achieved by analogy with a stereotypical producer-consumer pattern, without being concerned with the details of the underlying mechanisms.

Limitations. The present study has been conceived with an exploratory character, in order to gather preliminary insight into a range of aspects in connection with the learning of concurrent programming in the context of our school system. Of course, each such aspect would be worth a specific, more focused investigation, possibly involving larger student samples in a wider geographic area.

Implications for Educators. Mastery of basic concepts of concurrent programming is a cognitively demanding endeavor that, in order to nurture meaningful learning, requires much pedagogical effort and time spent on the subject.

As also pointed out by multiple students, teachers should be particularly careful to choose an appropriate set of examples of gradually increasing complexity. An additional issue worth being considered is the use of graphical/visual aids, especially to support the integration of spatial abilities into the learning process [4,24]. A range of existing tools has been reviewed, e.g., by Libert & Vanhoof [15]. Finally, the low correlation between self-confidence and performance suggests that more attention needs to be paid to students' metacognitive skills [18], possibly by offering them "opportunities for empirical validation of their knowledge" and explicit instruction in this respect [19, p. 148].

6   Conclusions

In this paper we have presented the results of an exploratory investigation, carried out via a survey, addressing high school students' perception, proficiency and self-confidence when dealing with concurrent programming tasks. While the main implications of our findings are discussed in the previous section, appropriate decisions about the potential role of this subject in a high school context, and specifically in our school system, are bound to find a reasonable trade-off between two opposite poles: on the one hand, the relevance of the topic from a professional perspectives [9]; on the other, the high cognitive challenge [2] in light of learners' maturity and teaching time available to develop the subject.

Future Perspectives. Besides designing and planning more focused investigations to overcome the limitations mentioned above, a shorter-term goal could be to administer the current version of the survey in other technical high schools following heterogeneous approaches to the teaching of concurrent programming, with the aim of assessing the extent to which different instructional approaches influence students' perceptions and/or achievements. Further research could be devised in order to evaluate and compare the effectiveness of different graphical/visual tools to improve the understanding of thread-related concepts.

Appendix: Survey Questions

An easier-to-read version of the survey questions is also available online at the link:
http://nid.dimi.uniud.it/additional_material/thread_survey/thread_survey.pdf

Approach to threads

– In general, how would you rate the difficulty of the thread topic?
   4–grade Likert scale (1 = Not difficult – 4 = Very difficult)

– How would you rate your performance when managing threaded applications?
   4–grade Likert scale (1 = Not satisfied – 4 = Very satisfied)

– In your opinion, is it adequate the amount of time that the teacher spends to introduce the thread topic?
4–grade Likert scale (1 = Not adequate – 4 = Definitely adequate)

– In your opinion, are the examples and exercises that the teacher proposes to introduce the thread topic adequate?
4–grade Likert scale (1 = Not adequate – 4 = Definitely adequate)

– Rate the level of difficulty you typically encounter when dealing with the following thread issues. (Mark only one option per row)
Options: not known, usually simple, reasonably simple, difficult, very difficult.
Topics: Class definition, Object shared between threads, Distinguishing shareable vs. non–shareable data, Thread "Run" method definition, Starting a thread, Closing a thread, Choice of class methods, Identification of shared class methods, Understand thread life cycle, Dealing with thread state, Synchronization (in general).

– Rate the level of difficulty you encounter when using the following methods for managing the state of a thread. (Mark only one option per row)
Options: not known, usually simple, reasonably simple, difficult, very difficult.
Methods: start, stop, sleep, suspend, wait, yield, join, resume, notify, notifyAll, synchronized.

– Rate the level of difficulty you encounter when dealing with conditions between threads. (Mark only one option per row)
Options: not known, usually simple, reasonably simple, difficult, very difficult.
Operations: Read a shared resource, Write or modify a shared resource, Accidental resource sharing, Early release of a resource, Multiple Locks for the same resource, Missed protection of a shared resource, Synchronization of shared resources, Synchronization of methods that manage shared resources, Wait without wake–up notification (Notify).

Tasks
The code fragments formalized in Java for Task 1.a–d refer to the *Counter* class defined as follows:

```java
public class Counter {
    private int count = -1; // a negative value of count is interpreted as ``undefined''

    public synchronized int getCount() {
        while ( count < 0 ) {
            try {
                wait();
            } catch ( Exception e ) {}
        }
        return count;
    }

    public synchronized void setCounter( int initValue ) {
        if ( initValue >= 0 ) {
```

```
        count = initValue;
        notify();
    }
  }

  public synchronized void increment() {
    while ( count < 0 ) {
      try {
        wait();
      } catch ( Exception e ) {}
    }
    count = count + 1;
  }
} // Counter
```

Task 1.a Analyze the execution of the following code snippets (Fig. 4) for two separate threads, Thread-1 and Thread-2, operating on a shared instance x of the Counter class introduced above. The operations of each of the two threads are represented along opposite sides of the vertical axis, according to the time order (from top to bottom) in which the methods invoked in the instructions are executed; furthermore, no operations on x or i have been omitted in the reported flows. What are the output values printed during the execution of Thread-1? Mark only one option.

Options: i = 1, count = 1; i = 1, count = 5; i = 5, count = 5; i = 6, count = 6; The result cannot be predicted because there are several possibilities.

Task 1.b Analyze the execution of the following code snippets (Fig. 5) for two separate threads, Thread-1 and Thread-2, operating on a shared instance x of the Counter class introduced above. The operations of each of the two threads are represented as described in question Task 1.a. What are the output values printed during the execution of Thread-1? Mark only one option.

Options: i = 1, count = 1; i = 1, count = 5; i = 5, count = 5; i = 5, count = 6; The result cannot be predicted because there are several possibilities.
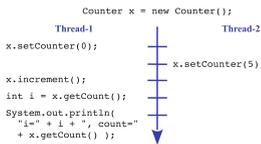
Fig. 4. Task 1.a.
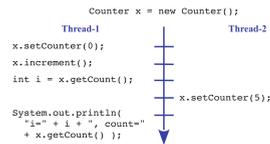
Fig. 5. Task 1.b

Task 1.c Analyze the execution of the following code snippets (Fig. 6) for two separate threads, Thread-1 and Thread-2, operating on a shared instance x of the Counter class introduced above. The operations of each of the two threads are represented as described in question Task 1.a. What are the output values printed during the execution of Thread-1? Mark only one option.

Options: i = 0, count = 0; i = 5, count = 0; i = 6, count = 0; i = 6, count = 6; The result cannot be predicted because there are several possibilities.

Task 1.d Analyze the execution of the following code snippets (Fig. 7) for two separate threads, Thread-1 and Thread-2, operating on a shared instance x of the Counter class introduced above. The operations of each of the two threads

are represented as described in question Task 1.a. What are the output values printed during the execution of Thread-1? Mark only one option.

Options: i = 0, count = 0; i = 5, count = 6; i = 6, count = 6; i = −1, count = 6; The result cannot be predicted because there are several possibilities.

Fig. 6. Task 1.c

Fig. 7. Task 1.d

Task 1: self-confidence level With regard to the previous questions (Task 1.a–d), rate your degree of confidence in the correctness of the solutions you have chosen on a scale from 1 to 4.

4–grade Likert scale (1 = Not confident at all – 4 = Fully confident)

Task 2: Consider the classes defined below (Fig. 8) and assume to start the program through the *main* method of the Task2 class. Which of the proposed sequences will be printed at the end of the execution? Mark only one option.

Options: P3P7P5; P3PP7PP5P; PP3P5P7; PP3PP7PP5; PPP375; PPPP PP375; The program hangs in a deadlock; The result cannot be predicted because there are several possibilities.

Task 2: self-confidence level With regard to the previous question (Task 2), rate your degree of confidence in the correctness of the solution you have chosen on a scale from 1 to 4.

4–grade Likert scale (1 = Not confident at all – 4 = Fully confident)

Task 3: Within a class describing the implementation of a shared resource, which of the following methods' definitions (Fig. 9) can help to avoid conflicts in the management of the resource itself?

Fig. 8. Task 2

Fig. 9. Task 3. Equivalence: *Select all applicable items.*

Task 3: self-confidence level With regard to the previous question (Task 3), rate your degree of confidence in the correctness of the solution you have chosen on a scale from 1 to 4.

4–grade Likert scale (1 = Not confident at all – 4 = Fully confident)

Task 4: Consider an instance of the *Bouncer* class defined below. The synchronization modes of the *from1to2* and *from2to1* methods can lead to deadlock situations.

Which of the following workarounds will fix the code to prevent the occurrence of a deadlock (while still ensuring proper synchronization)?

Mark only one option.: delete all synchronized; eliminate nested synchronized; drop synchronized by either method; drop the outer synchronized from one of the methods and the nested one from the other; transform nested synchronized into sequenced synchronized (one after the other rather than one within the other); reverse seq1 and seq2 in all synchronized constructs; none of the previous solutions.

```java
public class Bouncer {
  private Vector<Integer> seq1;
  private Vector<Integer> seq2;

  public Bouncer( Vector<Integer> seq1, Vector<Integer> seq2 ) {

    this.seq1 = seq1;
    this.seq2 = seq2;
  }

  public void from1to2() {
    synchronized ( seq1 ) {
      if ( seq1.size() == 0 ) {
        try {
          seq1.wait();
        } catch ( Exception e ) {}
      }
      int item = seq1.elementAt(0);
      seq1.removeElementAt(0);
      synchronized ( seq2 ) {
        seq2.add( item );
        seq2.notify();
      }
    }
  }

  public void from2to1() {
    synchronized ( seq2 ) {
      if ( seq2.size() == 0 ) {
        try {
          seq2.wait();
        } catch ( Exception e ) {}
      }
      int item = seq2.elementAt(0);
      seq2.removeElementAt(0);
      synchronized ( seq1 ) {
        seq1.add( item );
        seq1.notify();
      }
    }
  }
} // Bouncer
```

Task 4: self-confidence level With regard to the previous question (Task 4), rate your degree of confidence in the correctness of the solution you have chosen on a scale from 1 to 4.–grade Likert scale (1 = Not confident at all – 4 = Fully confident)

Possible help tools

– Have you ever thought about a graphical representation of thread working principles, in order to ease its understanding? –grade Likert scale (1 = Never – 4 = Often)
– Do you think a graphical representation of how threads work could be effective in improving your understanding? –grade Likert scale (1=Not effective – 4 = Very effective)
– How would you rate the following graphing tools in the context of threads? (Mark only one option per row) : I don't know this type of representation, not very useful, partially useful, quite useful, very helpful.: flow–charts, Petri nets, finite state automata, Cartesian diagrams as a function of time, Unified Modeling Language (UML), Holt graphs, block diagrams.

Final open question

– What would you suggest to make the lessons on threads more interesting and clearer? (Open answer)

References

1. Ben-Ari, M., Kolikant, Y.B.D.: Thinking parallel: the process of learning concurrency. SIGCSE Bull. **31**(3), 13–16 (1999)
2. Brođanac, P., Novak, J., Boljat, I.: Has the time come to teach parallel programming to secondary school students? Heliyon **8**(1), e08662 (2022)
3. Bruce, K.B., Danyluk, A., Murtagh, T.: Introducing concurrency in CS1. In: Proceedings of the 41st ACM Technical Symposium on Computer Science Education, SIGCSE 2010, pp. 224–228. ACM, New York, NY, USA (2010)
4. Buckley, J., Seery, N., Canty, D.: Spatial cognition in engineering education: developing a spatial ability framework to support the translation of theory into practice. Eur. J. Eng. Educ. **44**(1–2), 164–178 (2019)
5. Choi, S.E., Lewis, E.C.: A study of common pitfalls in simple multi-threaded programs. In: Proceedings of the 31st SIGCSE Technical Symposium on Computer Science Education, SIGCSE, pp. 325–329. ACM, New York, NY, USA (2000)
6. Conte, D.J., de Souza, P.S.L., Martins, G., Bruschi, S.M.: Teaching parallel programming for beginners in computer science. In: 2020 IEEE Frontiers in Education Conference (FIE), pp. 1–9 (2020)
7. Fekete, A.D.: Teaching students to develop thread-safe java classes. In: Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE 2008, pp. 119–123. ACM, New York, NY, USA (2008)
8. Fernández, A., Fernández, C., Miguel-Dávila, J.A., Conde, M.A., Matellán, V.: Supercomputers to improve the performance in higher education: a review of the literature. Comput. Educ. **128**, 353–364 (2019)
9. Gardner, W.B.: Should we be teaching parallel programming? In: Proceedings of the 22nd Western Canadian Conference on Computing Education, WCCCE 2017, ACM, New York, NY, USA (2017)
10. Hartley, S.J.: Alfonse, wait here for my signal!. In: Proceedings of the 30th SIGCSE Technical Symposium on Computer Science Education, SIGCSE 1999, pp. 58–62. ACM, New York, NY, USA (1999)

11. John, D.J., Thomas, S.J.: Parallel and distributed computing across the computer science curriculum. In: 2014 IEEE International Parallel & Distributed Processing Symposium Workshops, pp. 1085–1090 (2014)

12. Joint Task Force on Computing Curricula: Association for Computing Machinery (ACM) and IEEE Computer Society: Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science. Association for Computing Machinery, New York, NY, USA (2013)

13. Ko, Y., Burgstaller, B., Scholz, B.: Parallel from the beginning: the case for multicore programming in the computer science undergraduate curriculum. In: Proceedings of the 44th ACM Technical Symposium on Computer Science Education, SIGCSE 2013, pp. 415–420. ACM, New York, NY, USA (2013)

14. Kolikant, Y.B.D.: Learning concurrency: evolution of students' understanding of synchronization. Int. J. Hum.-Comput. Stud. **60**(2), 243–268 (2004)

15. Libert, C., Vanhoof, W.: Survey of software visualization systems to teach message-passing concurrency in secondary school. In: Bajo, J., et al. (eds.) PAAMS 2017, vol. 722, pp. 386–397. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-60285-1_33

16. Lönnberg, J., Berglund, A., Malmi, L.: How students develop concurrent programs. In: Proceedings of the 11th Australasian Conference on Computing Education, ACE 2009, vol. 95. pp. 129–138. Australian Computer Society, Inc. (2009)

17. Luxton-Reilly, A., Simon et al.: Introductory programming: a systematic literature review. In: Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE 2018, pp. 55–106. ACM, New York, NY, USA (2018)

18. Mani, M., Mazumder, Q.: Incorporating metacognition into learning. In: Proceedings of the 44th ACM Technical Symposium on Computer Science Education, SIGCSE 2013, pp. 53–58. ACM, New York, USA (2013)

19. Murphy, L., Tenenberg, J.: Do computer science students know what they know? A calibration study of data structure knowledge. In: Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education, ITiCSE 2005, pp. 148–152. ACM, New York, USA (2005)

20. Rifkin, A.: Teaching parallel programming and software engineering concepts to high school students. In: Proceedings of the 25th SIGCSE Symposium on Computer Science Education, SIGCSE 1994, pp. 26–30. ACM, New York, NY, USA (1994)

21. Rivoire, S.: A breadth-first course in multicore and manycore programming. In: Proceedings of the 41st ACM Technical Symposium on Computer Science Education, SIGCSE 2010, pp. 214–218. ACM, New York, NY, USA (2010)

22. Scapin, E., Mirolo, C.: An exploration of teachers' perspective about the learning of iteration-control constructs. In: Pozdniakov, S.N., Dagienė, V. (eds.) ISSEP 2019. LNCS, vol. 11913, pp. 15–27. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-33759-9_2

23. Torbert, S., Vishkin, U., Tzur, R., Ellison, D.J.: Is teaching parallel algorithmic thinking to high school students possible? One teacher's experience. In: Proceedings of the 41st ACM Technical Symposium on Computer Science Education, SIGCSE 2010, pp. 290–294. ACM, New York, NY, USA (2010)

24. Wai, J., Lubinski, D., Benbow, C.P.: Spatial ability for STEM domains: aligning over 50 years of cumulative psychological knowledge solidifies its importance. J. Educ. Psychol. **101**(4), 817–835 (2009)
25. Zhu, J., Alderfer, K., Smith, B., Char, B., Ontañón, S.: Understanding learners' problem-solving strategies in concurrent and parallel programming: A game-based approach (2020). arXiv.org (ARXIV2005.04789)

Robotics and Unplugged Modalities

Combining Models to Orchestrate an Instructional Scenario Fostering Computational Thinking in Educational Robotics

Frankie Dubois, Stephanie Burton, Patrick Wang(✉), and Morgane Chevalier

University of Teacher Education, Lausanne, Switzerland
p49499@etu.hepl.ch,
{stephanie.burton,patrick.wang,morgane.chevalier}@hepl.ch

Abstract. Computational thinking (CT) is often defined as multifaceted which, on the one hand, allows researchers to embrace its complexity but, on the other hand, blurs the possibilities of its teaching. Although many models shed light on the multiple dimensions of CT, few studies investigate the benefits of combining such models when a teacher orchestrates in-class activities aiming at developing students' CT. This position paper aims to fill this gap by describing and analysing how a teacher can base the orchestration of a pedagogical scenario on three different models: Komis et al.'s model to design ER activities in co-creative problem solving, Sentance et al.'s PRIMM model to scaffold the students' tasks, and Chevalier et al.'s CCPS model to unscaffold the learning activities.

Keywords: Educational Robotics · Computational Thinking · Teacher's Guidance

1 Introduction

The construction of knowledge in computer science (CS) teaching and, in particular, during educational robotics learning activities (ERLA), can be based on three learning theories, namely constructivism [11], constructionism [10] and socio-constructivism [17]. These theories may be too abstract for teachers to be put in practice, limiting the number of studies available on their impacts on the students' learning processes. The scientific literature is nonetheless rich in recommendations on specific aspects of the construction of the knowledge at stake, for example, during an ERLA aimed at fostering student's computational thinking (CT) skills. In this respect, some studies [5,6] highlighted teaching and learning strategies that enable teachers to structure activities and encourage students to develop solutions to problem situations that have been thought out in advance whilst avoiding trial-and-error approaches that are generally unproductive and discouraging [2]. Such studies have the advantage of providing concrete

evidence of how CT is developed in the classroom but they only take into account part of the complexity involved in this thought processes. Yet teachers must take into account all this complexity and address it in a planned way. This is where comes in one's classroom orchestration expertise i.e. the ability to design and conduct "multi-plane scenarios under multiple constraints" [8].

Thus, to support teachers in their lesson planning, our research question is the following: When developing and implementing educational robotics scenarios, how can teachers combine different models validated by scientific research to foster students' computational thinking? In order to achieve this, a structure of the pedagogical scenario can be considered with reference to the scenario-based approach in educational robotics [9]. This approach will be reinforced by implementing the strategies developed in the PRIMM model [15] and in the CCPS model [6].

This paper has the following structure: in Sect. 2, we present our theoretical framework i.e., the three CT dimensions and three models to foster each of these facets. In Sect. 3, based on the combination of these three models, we expose the design of our pedagogical scenario to foster CT in ERLA. Subsequently, we highlight in Sect. 4 the relations between our theoretical model and the pedagogical scenario in order to justify its design. Finally, we conclude in Sect. 5.

2 Theoretical Framework

2.1 The CT Concept and Its Three Main Dimensions

According to Brennan and Resnick [4], CT is made up of three facets:

- **Computational perspectives**, which consist of cross-disciplinary abilities that are not characteristic of problem-solving within the framework of computer science. For example, these abilities can be to identify or generalize a problem, to model, generate ideas and communicate them.
- **Computational concepts**, which encompass the notions of computer science called upon during the learning activity and therefore, according to [13], notions linked to both the machine to be programmed (e.g., knowledge of the components of the robot used) and its programming language (e.g., knowledge of syntax and semantics, but also the knowledge of the interface used to program the robot).
- **Computational practices**, which refer to the skills required in the actual act of programming. This involves being able to decompose a problem or to modularize it, to test and evaluate a solution or to debug it, as well as working iteratively towards a satisfying solution.

CT comprises complex thought processes [1,18] that cannot be achieved directly. By conferring the status of competence to CT [7], it is then justified to approach CT in act (for example, via collaborative and creative co-construction) and in a situated way (for example, within a problem-solving situation in educational robotics). In order to ensure that we develop the full complexity of CT

(and not just one of its three dimensions), it therefore makes sense to use models validated in the literature and aimed at the same intentions formulated through the three CT dimensions according to Brennan and Resnick [4].

2.2 Scenario-Based Approach for Educational Robotics

The Scenario-Based Approach is a model designed to structure the ER activities for co-creative problem solving. Based on a constructivist-constructionist approach, it proposes a sequence of five activities to support the planning and orchestration of ER in K-12 education. This "ensure a progressive level of guidance towards the consolidation of the knowledge building process. The guidance is based on the scaffolding strategies of the Zone of Proximal Development (ZPD) [17]." [9, p.162]. In their paper, the authors identify the diversity of the ER activities that could be used in the first three activities of the scenario and claim that the last two activity types can be integrated within the first three. We therefore only describe the first three types of activity below.

Preparatory Activities: As the name suggests, this type of activity is designed to prepare learners for the ERLA: presentation of objectives, reminder of what is known about programming and robotics, presentation of the robotic tool used in the scenario. At this stage, however, students are not expected to manipulate the robot. Examples of this type of activity comprise lecture-based introduction to robotics or classroom debate about robotics.

Activities for Building Initial Knowledge: This second type of activity is designed to guide students in the use and manipulation of the educational robot leveraged in the scenario. The teacher plays a very important role at this point: to enable students to acquire the knowledge related to the robot's components and to its programming interface. Examples of this type of activity include individual guided activities or collaborative guided activities.

Activities for the Consolidation of Acquired Knowledge: This third type of activity is designed to enable students to design, manipulate and interact with their peers in a problem-solving situation. This gives students more responsibility and allows them to consolidate the knowledge built up in the previous two steps. Examples of this type of activity include individual or collaborative engineering problem, co-creative project-oriented robotic challenges.

2.3 The PRIMM Model

PRIMM [16] is a model designed to structure programming activities that avoid common difficulties found in the literature and, more specifically, the many challenges that students face when writing code. The approach followed by PRIMM is based on Vygotsky's Zone of Proximal Development [17] and consists in scaffolding the students' activities by proposing tasks that gradually progress from reading code to writing code.

PRIMM is an acronym which stands for Predict, Run, Investigate, Modify, Make. In the Predict phase, students are shown a piece of code and must work out (alone or in small groups) its outcome. In this stage of the model, students are asked to gather their code reading skills to figure out the result of the execution of the program. In the Run phase, the teacher executes the program so that students can confront their predictions with the actual outcome. In the Investigate phase, the teacher can question students regarding potential errors during the Predict phase and tries to explain these errors, or introduce new concepts to the class. The Investigate phase actually relies on the Block Model [14] to question the students' understanding of specific parts of the program, helping the acquisition or consolidation of programming knowledge. In the Modify phase, students are asked to work on the same program they had during the Predict phase and to modify it to adapt its execution based on a new set of (very similar) instructions. Finally, in the Make phase, students are asked to write a program from scratch by reusing what they have learnt throughout the process.

2.4   The CCPS Model

The CCPS model [6] is an instrument for planning and assessing whether the ERLA specifically promotes CT skills. In creative computational problem solving (CCPS), five phases (plus one to identify when the student is off-task) have been identified. All these phases are illustrated in Fig. 1. The remaining part of this section provides only a brief description of the model. For more information, we refer the readers to the original paper [6].

The first three phases of CCPS focus on three facets of CT (or the "computational perspective" dimension according to [4]): understanding the problem, generating ideas and formulating the robot's behavior. The other two phases refer to the "computational practices" dimension (ibid.): the fourth phase describes the creation of executable code (programming) by the robot, and the fifth phase focuses on executing the code to evaluate the solution. The "computational concepts" dimension (ibid.) is not identified in this model, as it is considered a prerequisite for the problem-solving task.

According to [6], during a CCPS in ER, students often rely on a trial-and-error strategy that is made possible by immediate feedback from the robots (for example, a LED activating on the robot to indicate to its user that the robot's infrared sensor has captured some information). However, this informational feedback can become less pedagogically relevant when students use it to promote task completion to the detriment of developing learning strategies [2]. For this reason, [6] suggests temporarily blocking access to the programming phase so as to promote the previous three phases of CCPS linked to the "computational perspective" (see the red "stop" sign in Fig. 1). In this way, the teacher's intervention can fade away (fading of scaffolding) during the problem-solving phase.
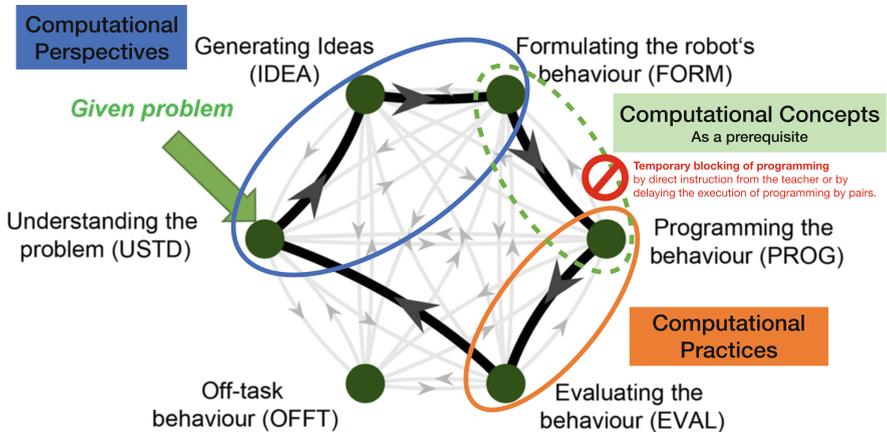
Fig. 1. The CCPS model [6] with six observable, interconnected phases (see grey arrows) and the three CT dimensions, marked in blue, green and orange. (Color figure online)

3   Design of the Instructional Scenario

Based on the state of the art and on the three models identified in Sect. 2, we designed the instructional scenario illustrated in Fig. 2 whose objective is to foster all three dimensions of CT in the context of educational robotics in primary school, for children aged 9–12 year old. The instructional scenario comprises eight activities in three different teaching stages, the latter referring to the three stages in the scenario-based approach [9] presented in Sect. 2.2. Activities #1 to #3 are "unplugged" activities [3], i.e. without the use of robots or any computing device. The following activities, on the other hand, are in plugged mode, and therefore include robots. The robot used in this proposal is the Blue-Bot[1].

The following description of the instructional scenario is also supported by the feedback from a teacher who co-designed and implemented it in a middle school classroom in Switzerland, with 10–12 year old pupils. While the design itself has not been rigorously evaluated, both the teacher and his pupils' feedback after the implementation of this scenario show that the combination of the three models selected is relevant. Indeed, the instructional scenario constructed in this way enabled the teacher to become aware of the stages to be respected in the construction of knowledge (in this case, the three CT dimensions) and to visualise more easily when he should or should not intervene with his pupils. It also enabled students to become more autonomous as the activities progressed and to develop important cross-curricular skills such as collaboration and creative thinking.

---

[1] https://www.tts-international.com/blue-bot-bluetooth-programmable-floor-robot/1015269.html.

Fig. 2. Description of the instructional scenario composed of eight activities divided in three stages. (Color figure online)

3.1   Phase 1: Preparatory Activities

As shown in Fig. 2, the first phase concerns preparatory activities and is made up of Activity #1 and Activity #2, described hereafter.

Activity #1 aims to engage students in the subject of robotics, and thus introduces it. The chosen working method is a class debate. This approach enables all students to get involved in the subject. The aim is to define what a robot is, so the teacher successively asks students the following questions: "What is a robot?", "What does it do?", "What does it need to function?" For each of these questions, students write a keyword anonymously on a piece of paper. The papers are then collected by a classmate, who reads them while the teacher notes them on the board. When they are pooled on the board, the students are asked if any of the words could be grouped together. This pooling then generates a class discussion and debate for each question.

Activity #2 aims for students to understand that the robot machine executes a program defined in a univocal language understood by both the programmer and the machine. In order to make them aware of this, students work in freely formed pairs. First, Student A is given the following instruction in front of the whole class: "Using only your voice, guide your blindfolded classmate to the classroom library. As his vocabulary is limited to verbs, you can't name the various obstacles". So, initially, Student B is blindfolded and the other students modify the configuration of the classroom tables to create obstacles. Student A gives instructions to Student B to solve the given problem (i.e., enables him to progress to the library while avoiding the obstacles). The other students in the class observe and identify what does and does not solve the problem. The pooling of these identifications then leads to the emergence of the need for a shared, unambiguous language. Finally, all the students in the class pair up and carry out the activity in turn, to test this need.

3.2   Phase 2: Building Initial Knowledge

As shown in Fig. 2, the second phase concerns the building of initial knowledge and is made up of four activities (#3, #4, #5 and #6).

Activity #3 involves reading a program and predicting its outcome. For this purpose, the chosen working method is mainly individual work. Sheet 1 (Fig. 3, left) is distributed to each student and the following question is asked: "Following the instructions in the program, can you tell which square the robot should arrive in?" Individually, students read the program and trace the robot's path and then compare their results with their neighbors. In the end, they try to identify similarities and possible errors. After pooling their results, they make a joint prediction of the outcome.
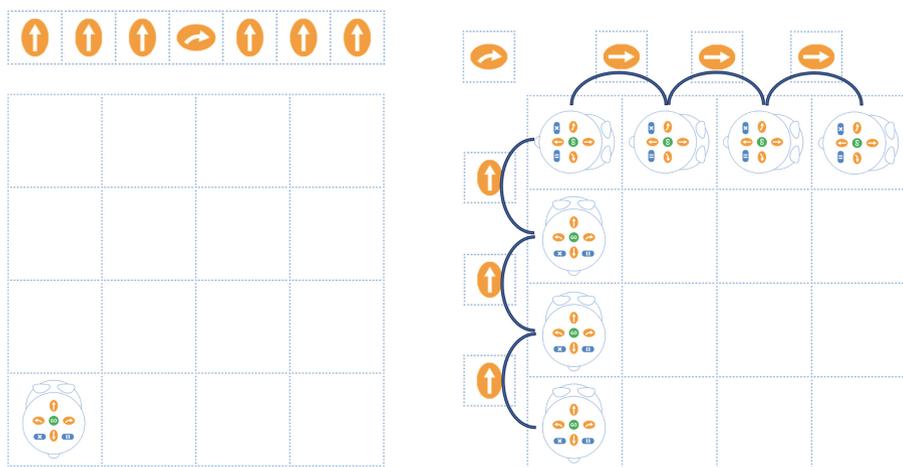
Fig. 3. Left: Example of a Predict task. Right: Guidance and pooling for the Investigate phase. (Color figure online)

Activity #4 aims to have a robot run the program of activity #3 and observe/investigate the results. The execution is carried out as many times as necessary to understand each step of the program. Again, students work in freely-formed pairs so as to encourage discussion when observing the results. With this plugged activity (at least for the "run" part), we need to prepare the following materials for each pair: a mat with 16 squares (15 cm by 15 cm), one Blue-Bot robot. The instructions to be given are as follows: "First, enter the program given on the robot and press the GO button to make the robot execute your program. Then, investigate each of the steps performed by the robot". A pooling of the investigation crystallizes the robot's programming language. The teacher's guidance can be based on the illustration on the right of Fig. 3.

Activity #5 aims to modify the program of activity #3. The working method chosen is once again in pairs, but the pairings must be different from the previous activity. The material is the same as in Activity #3, but now with

Sheet 2 (Fig. 4, left). The instructions to be given are as follows: "The robot's starting position has changed. How can this program be modified so that the robot arrives at the same position as before?" Paper programming cards are then used to support communication within the pairs. A final pooling of the results reveals both the difficulties encountered by the pairs and the strategies adopted to overcome them.

Activity #6 involves writing a program. The chosen working method is evenly matched pairs (in the same near-development zone). The materials required for each pair is the same as in Activity #4, but now with Sheet 3 (Fig. 4, right). The instructions to be given are as follows: "Create a program that allows the Blue-Bot robot to arrive at the garage". Students solve the given problem collaboratively, with the teacher intervening more with pairs who still need guidance.
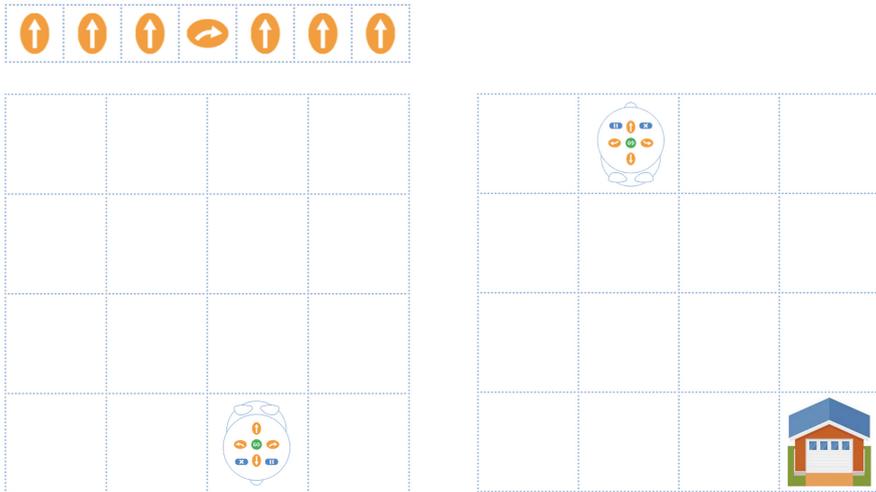
Fig. 4. Left: Example of a Modify activity. Right: Example of a Make activity. (Color figure online)

3.3   Phase 3: Consolidation of Acquired Knowledge

As shown in Fig. 2, the third and last phase concerns the consolidation of the acquired knowledge and is made up of Activity #7 and Activity #8.

Activity #7 involves creative computational problem-solving. The chosen working method is freely-formed pairs. The materials required for each pair are as follows (Fig. 5, left): a mat with nine green squares (15 cm by 15 cm), one Blue-Bot robot, a pencil-case for Blue-Bot, paper programming cards, paper and pencil. The instruction is: "The robot must mow the lawn symbolized by the nine green squares. The problem is considered solved if a pencil mark can be found

in each square. You have 20 min to solve this problem but with the following constraint: during the first five minutes, you cannot execute the program (by pressing the robot's Go button), you will be able to do so during the next five minutes; then, you will again be prevented from running the program for five minutes; finally, you will be able to do so during the last five minutes". The pooling of all the possible program scripts is held to validate the problem solution. A collective discussion should help identify the benefits of the constraints experienced during this resolution. The teacher explains the need to communicate within the group to clearly formulate the behavior of the robot to be programmed.
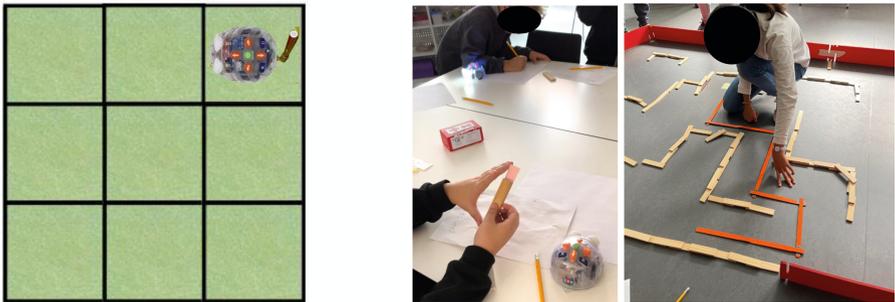
Fig. 5. Left: Material used for the lawnmower activity. Right: Materiel used for the "Theseus and the Minotaur" activity. (Color figure online)

Activity #8 involves creative computational problem-solving in a contextualized problem: Theseus and the Minotaur. The chosen working method is evenly matched pairs. The materials required for each pair are as follows (see Fig. 5, right): a maze (2 m by 3 m) located in another room, 1 Blue-Bot robot, paper programming cards, paper and pencil, conventional (ruler) and non-conventional (chablon) measuring tools. The instruction is: "The robot must get out of the maze (without destroying the walls) by executing the shortest possible program. You will only have two opportunities to come to the room where the maze is". Each member of a pair plays the role of either the programmer who programs the robot or the measurer who measures the number of necessary moves to reach the end of the maze. Finally, a pooling of all the possible program scripts is held to validate the shortest ones. A discussion brings out the different ways of thinking about computational problems in ER.

4   Design Rationale and Links with the Models

Orchestrating [8] an ERLA in the classroom involves, on the one hand, planning/designing it and, on the other, implementing it. Since the learning objective of this ERLA is CT, the three dimensions of CT [4] need to be taken into account: a) computational perspectives, b) computational concepts, and c) computational practices. The "Scenario-Based Approach for Educational Robotics"

model [9] enables structured planning of the construction of the knowledge at stake (in this case, CT). By proposing eight activities divided into three phases, we aim to support teachers in this process (see Fig. 6).
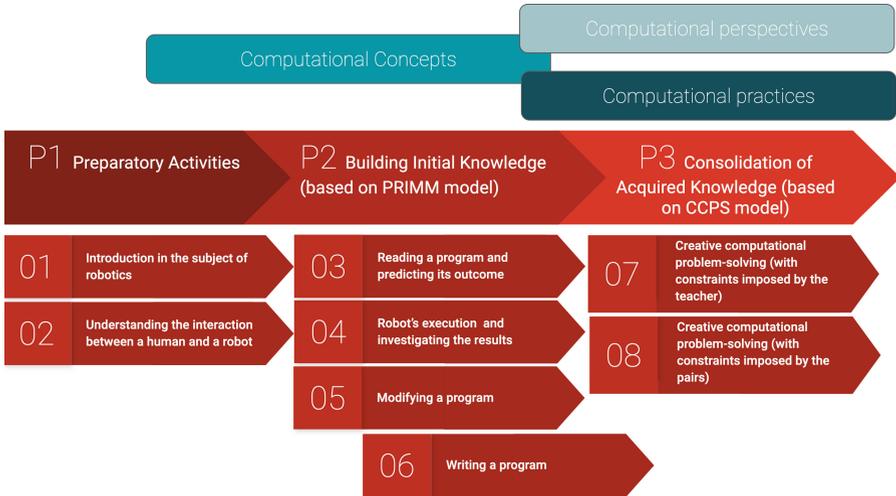
Fig. 6. Combining three models to orchestrate the three CT dimensions. (Color figure online)

Firstly, in phase 1, activities #1 and #2 are unplugged, so as to not burden the students' cognitive load with too much new knowledge (such as "computational concepts" like knowing the robot's components and programming interface). The priority here is to engage the students and challenge their initial representations. This phase also enables us to establish the principles of collaboration within the class. This basis will then be used as a lever during problem-solving in phase 3.

Secondly, in phase 2, we focus on building initial knowledge about CT. In this case, as identified by [6], the "computational concepts" dimension is a prerequisite for computational problem-solving (see dashed line in Fig. 1) since it must be possible to call on it when formulating the behaviour of the to be programmed robot and when programming it. To ensure the acquisition of these concepts, and more specifically of the programming language used by the robot, activities #3 to #6 are based on the PRIMM model [15] to support this learning through the following of five tasks: Predict, Run, Investigate, Modify, and Make. This last activity is a problem-solving task that indicates the CT competence level at this point in the scenario (end of phase 2). In this respect, we consider this to be a breaking point in the teacher's guidance. Up to this point, the teacher provided guidance and help to his students, leveraging the PRIMM model to scaffold the students' activities towards becoming more competent in the "computational concepts" dimension of CT. However, at this point in the scenario,

the students should have acquired a certain amount of autonomy regarding both the concepts and material in use, suggesting that a transfer of responsibility (from teacher to students) for learning [12] can occur. As a direct consequence, the teacher can now adapt and reduce his interventions, leaving the door open to the unscaffolding of learning activities.

Finally, in phase 3, higher-level learning should be encouraged [9]. With regard to CT, the aim is to reinforce the "computational practices" dimension while ensuring that the "computational perspectives" dimension takes root. This means proposing CCPS situations and planning a block to encourage all the phases of CT involved in the CCPS model, hence preventing students from going back and forth between programming and evaluating (see orange circle in Fig. 1).

The CCPS model is thus implemented in activities #7 and #8 with the goal of preventing unproductive trial-and-error strategies from occurring. For example, in activity #7, students are forced to work first on generating ideas and formulating expected behaviors without having any access to the programming environment (thus developing the "computational perspective" dimension of CT). After a few minutes, students can engage in the "computational practice" dimension by implementing and evaluating their solutions. This constraint is enforced by the teacher in activity #7 but then left to the responsibility of students during activity #8 (an additional rule can be added, stating that students can only test their solutions twice). This transfer of responsibility eventually participates in the unscaffolding strategy carried out in this third phase of the scenario.

An expectation of the CCPS model use is that students will engage in a virtuous cycle (as represented by the black arrows in Fig. 1) instead of getting stuck in the PROG-EVAL loop. With this structure of an instructional scenario, students should have developed all three facets of CT and, in the meantime, improved their CT skills since [6] shows that engaging in such a cycle seems to be beneficial with regards to the actual learning outcomes.

Finally, as indicated in Sect. 3, the scenario outilned was implemented by a teacher. His feedback is in line with the effects expected in this study. However, these effects still need to be measured more formally.

5   Conclusion and Future Work

In this position paper, we argued for the combination of three research models from the literature (scenario-based approach [9], PRIMM [16], and CCPS [6]) to support the design and orchestration of instructional scenarios aimed at fostering CT in ERLA. We back this claim by highlighting how these models cover each and every facet of CT as described in [4].

We also identify three possible leads for future work. First, we hope to evaluate how both expert and novices teachers implement this scenario in their classrooms and how it helps them plan and orchestrate the learning activities. Second, we wish to study the progression from a sequential-based robot (e.g., Blue-Bot) towards an event-based one (e.g., Thymio). And finally, we plan to assess the effects of such instructional scenarios on the students' learning outcomes regarding all three CT dimensions.

References

1. Aho, A.V.: Computation and computational thinking. Comput. J. **55**(7), 832–835 (2012)
2. Antle, A.N.: Exploring how children use their hands to think: an embodied interactional analysis. Behav. Inf. Technol. **32**(9), 938–954 (2013)
3. Bell, T., Vahrenhold, J.: Cs unplugged-how is it used, and does it work? Adventures between lower bounds and higher altitudes: essays dedicated to Juraj Hromkovič on the occasion of his 60th birthday, pp. 497–521 (2018)
4. Brennan, K., Resnick, M.: New frameworks for studying and assessing the development of computational thinking. In: Proceedings of the 2012 Annual Meeting of the American Educational Research Association, Vancouver, Canada, vol. 1, p. 25 (2012)
5. Chevalier, M., et al.: The role of feedback and guidance as intervention methods to foster computational thinking in educational robotics learning activities for primary school. Comput. Educ. **180**, 104431 (2022)
6. Chevalier, M., Giang, C., Piatti, A., Mondada, F.: Fostering computational thinking through educational robotics: a model for creative computational problem solving. Int. J. STEM Educ. **7**(1), 1–18 (2020)
7. Chevalier, M.S.D.: Mediating computational thinking through educational robotics in primary school. Technical report, EPFL (2022)
8. Dillenbourg, P., Prieto, L.P., Olsen, J.K.: Classroom orchestration. Int. Handb. Learn. Sci. 180–190 (2018)
9. Komis, V., Romero, M., Misirli, A.: A scenario-based approach for designing educational robotics activities for co-creative problem solving. In: Alimisis, D., Moro, M., Menegatti, E. (eds.) Edurobotics 2016 2016. AISC, vol. 560, pp. 158–169. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-55553-9_12
10. Papert, S., Harel, I.: Situating constructionism. Constructionism **36**(2), 1–11 (1991)
11. Piaget, J.: Six Etudes de Psychologie. Genéve: Editions Gonthier (1964)
12. Van de Pol, J., Volman, M., Beishuizen, J.: Scaffolding in teacher-student interaction: a decade of research. Educ. Psychol. Rev. **22**, 271–296 (2010)
13. Romero, M., Lepage, A., Lille, B.: Computational thinking development through creative programming in higher education. Int. J. Educ. Technol. High. Educ. **14**(1), 1–15 (2017)
14. Schulte, C.: Block model: an educational model of program comprehension as a tool for a scholarly approach to teaching. In: Proceedings of the Fourth International Workshop on Computing Education Research, pp. 149–160 (2008)
15. Sentance, S., Csizmadia, A.: Computing in the curriculum: challenges and strategies from a teacher's perspective. Educ. Inf. Technol. **22**(2), 469–495 (2017)
16. Sentance, S., Waite, J., Kallia, M.: Teaching computer programming with PRIMM: a sociocultural perspective. Comput. Sci. Educ. **29**(2–3), 136–176 (2019)
17. Vygotsky, L.S.: Psychology and localization of functions. Neuropsychologia **3**(4), 381–386 (1965)
18. Wing, J.M.: Computational thinking. Commun. ACM **49**(3), 33–35 (2006)

Teachers' Knowledge in Informatics—Exploring Educational Robotics Resources Through the Lens of Textual Data Analysis

Gabriel Parriaux[1]([envelope]) [ORCID], Christophe Reffay[2] [ORCID], Béatrice Drot-Delange[3] [ORCID], and Mehdi Khaneboubi[4]

[1] University of Teacher Education, Lausanne, Switzerland
gabriel.parriaux@hepl.ch
[2] University Bourgogne Franche-Comté, Besançon, France
christophe.reffay@univ-fcomte.fr
[3] University of Clermont Auvergne, Clermont-Ferrand, France
beatrice.drot-delange@uca.fr
[4] CY Cergy Paris University, Paris, France
mehdi.khaneboubi@u-cergy.fr

Abstract. In this research, we are interested in the knowledge of primary and secondary teachers to teach informatics. Using pedagogical resources produced by them as a trace of their enacted Pedagogical Content Knowledge (ePCK), we perform Textual Data Analysis and Clustering to discover the topics they write about. Focusing on resources in educational robotics, we show that lexicon used by teachers is different depending on the robot they use. Reinert's clustering associates each robot with a separate cluster and a specific vocabulary. Multiple Correspondence Analysis (MCA) shows an opposition between lexicon found in resources using event-driven robots (Thymio and Ozobot) and in resources using sequential robots (Beebot and Bluebot). Event-driven robots tend to be more related with events and behavior notions, as sequential robots tend to be more related with first manipulations of an object and programming notions.

Keywords: PCK in informatics · enacted PCK · educational robotics · programming paradigms · Reinert's clustering · Textual Data Analysis · teacher education

1 Introduction and Context

This research constitutes the third part of a research project on teachers' PCK in informatics that has been conducted between 2021 and 2023. In the first article [4], we investigated educational resources produced by pre-service primary teachers to teach programming in France and Switzerland. The results showed

that the progression of activities and the choice of teaching methods were not completely thought through by future teachers.

In the second article [13], we focused on a subset of our initial educational resources including robots. We added to it a set of online resources on the same topic from more experienced teachers. We wondered whether the computing notions involved in the resources were the same in the case of event-driven and sequential robots. Having extracted from the corpus the lexicon that we identified as "Computer Science (CS)", the results showed that it was significantly different according to the type of robot and the experience of the teacher.

In this third step, we want to explore further the potential of Textual Data Analysis techniques in the field of computing education. Unlike our previous research in which we focused on a portion of the lexicon that we manually extracted from the corpus, what would happen if we considered *the entire lexicon* and base our analyses on it? We propose to perform a new investigation on the same corpus of educational robotics resources, but this time using another method of Textual Data Analysis: Reinert's clustering. Reinert's clustering is an unsupervised divisive clustering technique [14] that, when applied to a corpus of texts, is able to extract some of their main topics, giving an idea of the mental universe embraced by their authors.

Our research questions are

RQ1 *What are the main themes that emerge from the textual analysis of educational resources for CS?*

RQ2 *What kind of relation can be established between the lexicon and the type of robot used in the resources?*

RQ3 *Based on the analysis of the lexicon, what other relations can be seen between the clusters, grade levels, programming paradigms and teachers' expertise?*

2   Related Work

Several models have been proposed to represent the knowledge of teachers. Among them, Pedagogical Content Knowledge (PCK) presented by Shulman in 1986, is one of the most fruitful. Built in reaction to a context in the US in which specific knowledge of the subject matter seemed to be left apart in teacher education, PCK is defined by Shulman as "a pedagogical knowledge that goes beyond knowledge of subject matter per se to the dimension of subject matter knowledge for teaching" [16, p. 9].

The original model was enriched by Magnusson, Krajcik and Borko for the teaching of science [9] who considered 5 components of PCK: orientations toward science teaching, knowledge of the curriculum, knowledge of students' understanding, knowledge of assessment and knowledge of instructional strategies.

A community of researchers in science education was built around the PCK model and proposed in a second iteration the Refined Consensus Model (RCM) of PCK in 2017 [2], which presents three different domains of PCK: collective PCK (cPCK), personal PCK (pPCK) and enacted PCK (ePCK).

Enacted PCK (ePCK) is the teacher's knowledge in action, "the specific knowledge and skills utilized by an individual teacher in a particular setting, with a particular student or group of students, with a goal for those students to learn a particular concept, collection of concepts, or a particular aspect of the discipline" [2, pp. 83–84]. It is not restricted to the moment of teaching in class with the students and also includes the planning and reflecting on the instruction by the teacher. According to Henze and Barendsen [6], ePCK is the part of the personal PCK that is active at a certain moment during teaching. Its dynamic aspect makes it not so easy to elicit and observe. Researchers interested in studying the enacted PCK of teachers can rely on classroom observations, but can also consider a pedagogical resource prepared by a teacher as a trace of its enacted PCK. This is the choice we made in studying pedagogical resources of school teachers who teach programming, they constitute a trace of their planning activity, a part of their ePCK.

Despite the fact that Textual Data Analysis is not a mainstream method in the field of education [5], we could identify some research where computational methods were applied to text in the context of education. Some of them have methodological aims and explicitly attempt to show that it's possible to use computational techniques to analyze textual data in education [1,15]. Some research use written data, for example open-ended questions in questionnaires to teachers or students [11,18], curricula [11] or teacher resources [1]. Other research use transcription of oral data, for example interviews [15] or discourse in class [8,10].

Few researchers employ Reinert's clustering in the context of education [5], but there exists a good example related to science education where Reinert's clustering is used to analyze teachers' representations of the investigation procedure, confronting clustering of an open-ended question in a teachers' survey and clustering of the content of the curriculum in science [11].

In the field of computing education, we found even less research using Textual Data Analysis, but we can mention one study using topic modeling to analyze students reviews on computer science MOOCs [3] and another study using Clustering technique of CS1 students' programming codes to identify a group of students with difficulties [17].

3   Data Collection and Methodology

Data collected for our research has two origins: the first part consists of 59 educational resources produced by pre-service teachers during workshops about the teaching of informatics in France and Switzerland. Those resources, dedicated to activities in educational robotics, were part of a more important corpus composed during the first phase of our research. More details are given about them in [4]. The second part of the data consists of 61 online resources on the same topic from more experienced teachers.

Our resources consist of pedagogical scenarios serving as preparation for a school lesson with one or more activities: teachers describe the learning objectives of the lesson, the roles of teachers and pupils, a schedule and the material

needed. Some resources also contain feedback about the difficulties encountered, either anticipated or experienced. For the pre-service teachers, the resources produced were used to validate one of their academic courses and their form differed depending on their university: a written scenario in a PDF document of 2 to 15 pages for some of them; a PowerPoint presentation explaining the scenario in the context of an oral exam; a video presenting the scenario in the same context, from which we extracted and transcribed the audio.

In order to enrich our corpus, we decided to augment it with 61 pedagogical scenarios available online. We followed a formalized process to use a search engine with keywords that would let us discover different resources for each robot, filtered the first results to download only resources that we could clearly identify as pedagogical scenarios. The documents were written documents in PDF between 4 and 20 pages. We have less information about the authors of these online documents than we have for our pre-service teachers, but we categorize them as "experienced" because we could see going through the documents that they were often well elaborated and that some of them were clearly affiliated to an educational academy.

We are conscious that the format of our resources is quite different depending on the context and it can be considered as a limitation of our research. But at the same time, the main goal of the documents was the same: presenting the pedagogical scenario of a school activity with robots. They are all textual and contain a lexicon that we can valuably analyze with Textual Data Analysis methods.

Finally, our corpus is composed of a total of 120 educational resources in French describing activities in educational robotics created either by novices or by experienced teachers.
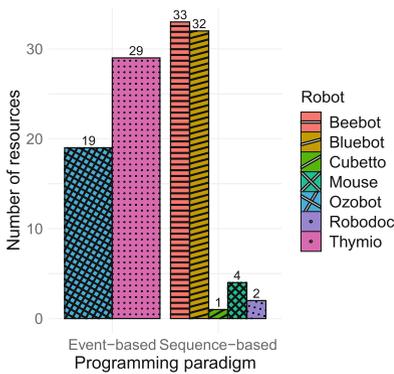
**Fig. 1.** Resources classified by type of robot and programming paradigm.
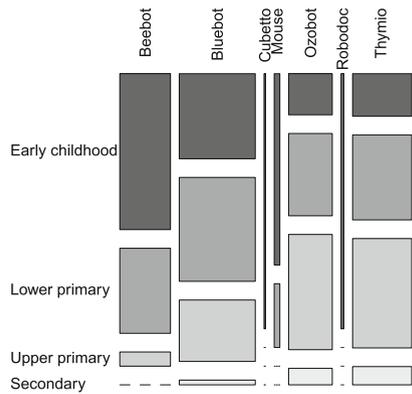
Fig. 2. Resources classified by type of robot and grade level.

Figure 1 shows the number of resources per robot in our corpus and their repartition between event-based and sequential. Event-based robots have sensors and actuators, they are mostly programmed in an event-based mode: various events are associated with specific behaviors; sequential robots do not have information from their environment and are programmed in a sequential mode: a sequence of instructions that are executed from a beginning to an end. Figure 2 presents the repartition of resources by robots and by grade level. We see that there are very few resources related to secondary education.

Processing of the data, pretreatment and analyses are performed in R [12]. The first step consists of extracting all the text from the resources. Having classified them according to categorical variables like the type of robot (sequential or event-driven), teachers' experience (novices or experienced), the grade levels and so on, we add those as metadata about the documents.

Before pretreatment, the whole corpus is composed of 248,813 occurrences and a vocabulary of 54,819 different words.

In order to perform Reinert's clustering, the corpus is split into segments of around 40 words, trying to maintain together words in complete sentences. We obtain 4,784 segments of text.

Then, we perform the lemmatization of the text, an operation that replaces every inflected form of a word into its unique normalized form. In the later process, forms that appear too few times in the corpus will be ignored in the analysis. Without lemmatization there is the risk to lose certain words where every inflected form would be counted as a different word, which is especially true for verbs in French. We use SPACYR, a wrapper for SPACY Python's library in R, with `fr_dep_news_trf`, a French transformer pipeline as a model, to process it.

Tokenization is then performed, which means the transformation of continuous text of the segments into individual tokens or words. The whole text is split into single tokens, with the space character as a natural separator. Punctuation, symbols and URLs are removed. We remove stopwords so that we can focus on notions which appear mostly through substantives or verbs. We use a custom list of 449 stopwords, as some of the common lists provided in R packages were rather too short (157 words for the standard) or too long (687 words for the list called `stopwords-iso`).

Finally, we compose an exclusion list along the classification process to remove words that are appearing in clusters and that make no sense for the analysis (for example abbreviations for school years, name of institutions or schools). It is composed of 116 words.

At the end of this pretreatment process, we obtain a matrix called *document-feature matrix* crossing 2,599 words and 4,784 segments which serves as a basis for the Reinert's clustering.

Reinert's clustering is an unsupervised divisive clustering that gets applied to the document-feature matrix. It means that it performs a bipartition of the segments iteratively. Segments are divided into two groups with the aim of building clusters that are as homogeneous as possible, keeping together the most similar

segments, and as heterogeneous as possible between them, with groups of segments as different as possible. Similarity is established through the presence of similar words in the segments using a Pearson's chi-squared test to compute the distance between the two clusters.

Reinert's method proposes a double clustering that crosses the results between two simple clustering performed with different minimal sizes of segments to obtain more robust classes. Trying various settings, it is up to the researcher to determine the number of clusters which seems the most relevant in terms of interpretation.

Having compared different analyzes, we decide to perform a double Reinert's clustering with a `min_segment_size = 10` for the first clustering and a `min_segment_size = 15` for the second. We keep eleven clusters. As is the case with double classification, no dendrogram and no visible hierarchy between the clusters is produced. We obtain a graph of the eleven clusters with a list of words that are strongly associated with them.

4   Results and Discussion

4.1   Results from Reinert's Clustering

A plot of the classification with the 11 clusters is displayed in Fig. 3, showing for each cluster the 30 tokens most significantly associated with. The number of unclassified segments is 821. It's an important number compared with the total of 4,784 segments (around 17%), but even though we decide not to force the classification of the remaining segments in the existing clusters, as this is reputed to potentially alter the homogeneity of the clusters.

Here is a list of the main clusters, with a title and some of the words that appear as strongly associated to—or overrepresented in—each cluster. An interface in the RAINETTE package lets us access the segments containing the words of a cluster easily, giving the necessary context to let us interpret better its meaning. The title is given according to our interpretation. Clusters 2, 3, 4 and 9 are small clusters with less than 200 segments and can be ignored.

- Cluster 1—Goals: *domain, grade, language, competence, objective, pedagogical, learning* (1,029 segments)
- Cluster 5—Movement: *erase, move forward, memory, left, right, turn, rotate, beebot* (203 segments)
- Cluster 6—Path: *path, route, cover, draw, time* (230 segments)
- Cluster 7—Sharing: *institutionalization, pupil, groups, collective, sharing* (531 segments)
- Cluster 8—Programming: *instruction, square, start, program, programming, check, bug, bluebot* (537 segments)
- Cluster 10—Events: *sensor, detect, behavior, object, event, explore, thymio* (543 segments)
- Cluster 11—Dancing: *movements, dance, tablet, code, app, ozobot* (515 segments)

Clusters size (NA = 821)

Cl. 1 n = 1029	Cl. 2 n = 172	Cl. 3 n = 13	Cl. 4 n = 85	Cl. 5 n = 203
domaine	tapis	ruban	pochette	effacer
cycle	cm	adhésif	direction	avancer
langage	grillager	ciseau	nappe	mémoire
compétence	image	colle	bâtonnet	gauche
spatial	région	crayon	carte	droite
espace	asseoir	kapla	appui	tourner
objectif	lettre	lego	prérequis	pivote
représentation	poisson	carton	envisager	touche
oral	album	plot	numérique86	pivoter
marqueur	transparent	choix	recontextualisation	reculer
démarche	comporter	imprimer	enchaîner	droit
apprentissage	petit	matériel	case	tromper
scientifique	ville	trio	répétition	ruche
pédagogique	animal	influencer	bande	beebot
communication	costume	défi	boucle	rituel
utiliser	sort	gros	quadriller	remettre
science	sac	document	factice	étiquette
repère	nord	quille	bluebot...	go
école	coccinelle	duo	changement	oublier
mobiliser	chiffre	stylo	arrivée	direction
numérique	case	pièce	test	allée
socle	porter	règle	grand	tourne
outil	poule	élève	quadrillage	rotation
technique	plénum	feutre	plastique	pivot
comprendre	grille	fiche	variable	épaule
mathématique	france	gris	beebot	case
viser	ferme	niveau	débrancher	falloir
français	contexte	numéro	instruction	appuyer
devoir	source	renverse	piste	quart
faire	costumer	3d	route	thymio

| Cl. 6 n = 230	Cl. 7 n = 531	Cl. 8 n = 537	Cl. 9 n = 105	Cl. 10 n = 543	Cl. 11 n = 515
nécessaire	mise	case	min	thymio	ozobot
chemin	commun	instruction	fleur	détecte	tablette
parcours	élève	cible	aller	capteur	danse
parcourir	fiche	arrivée	remobilisation	comportement	mouvement
chronométrer	collectif	élève	entraînement	tourne	cliquer
dessiner	enseignant	carte	consigne	lorsqu	pouvoir
tunnel	min	parcours	déroule	allume	dance
mettre	groupe	programmer	fromage	objet	feuillet
disposition	institutionnalisation	bluebot	binôme	bleu	charger
dessine	demande	atteindre	étiquette	jaune	forward
place	turn	programme	collectif	recule	code
vérifier	compléter	devoir	précédent	éteindre	carré
correspondre	colonne	interdire	étayage	vert	application
autocollant	straight	départ	colby	violet	app
temps	différent	erreur	manger	rouge	luminosité
tracer	activité	manquante	imiter	central	écran
modifier	séance	déplacement	groupe	appuyer	ozoblockly
noter	affiche	point	souris	clair	timeline
défi	réponse	étape	enseignant	détecter	statique
variable	rappeler	trajet	remarquer	allumer	calibrer
dispositif	découverte	trésor	rôle	bouton	cours
épais	hypothèse	barre	ruche	événement	possibilité
camarade	intérieur	vérifier	abeille	bruit	ios
inventer	présentation	difficulté	beebot	foncé	éditeur
feuille	pe	multiplicatif	diriger	activer	bloc
devoir	phase	économe	motricité	explorateur	clignotant
supplémentaire	recherche	économique	avancer	peureux	commencer
pouvoir	déplacement	arriver	réponse	mode	choisir
tester	gauche	quadrillage	mutualisation	devant	retourner
réaliser	droite	thymio	rappel	amical	élève

Fig. 3. Eleven clusters produced by Reinert's clustering. Lexicon displayed in French. Image produced in R with RAINETTE package.

4.2 Results from Multiple Correspondence Analysis

Once Reinert's clustering has been performed, we execute a Multiple Correspondence Analysis (MCA) to explore the relations between our clusters and the different variables associated to the pedagogical resources in our corpus.

A contingency table is constituted with eight categorical variables in columns and the 4,784 segments of texts as individuals in rows. The categorical variables associated with the segments are: *cluster* in which the segment is classified, *programming paradigm* of robots, *name* of robots, *level of expertise* of teachers and four *grade levels* of pupils concerned by the activities (early childhood, lower primary, upper primary and secondary). As our resources sometimes cover more than one grade level, we had to define grade levels as four separate variables. To gain in readability and avoid unnecessary information, segments concerning robots that rarely appear in our corpus are removed. Segments classified in the

small clusters 2, 3, 4 and 9 are also removed. We end with a contingency table of 4,216 rows and 8 columns.

The result of MCA can be seen in Fig. 4 for axes 1 and 2. Fifteen factors or axes are produced. First axis retains 22% of inertia and second axis 10.3%.
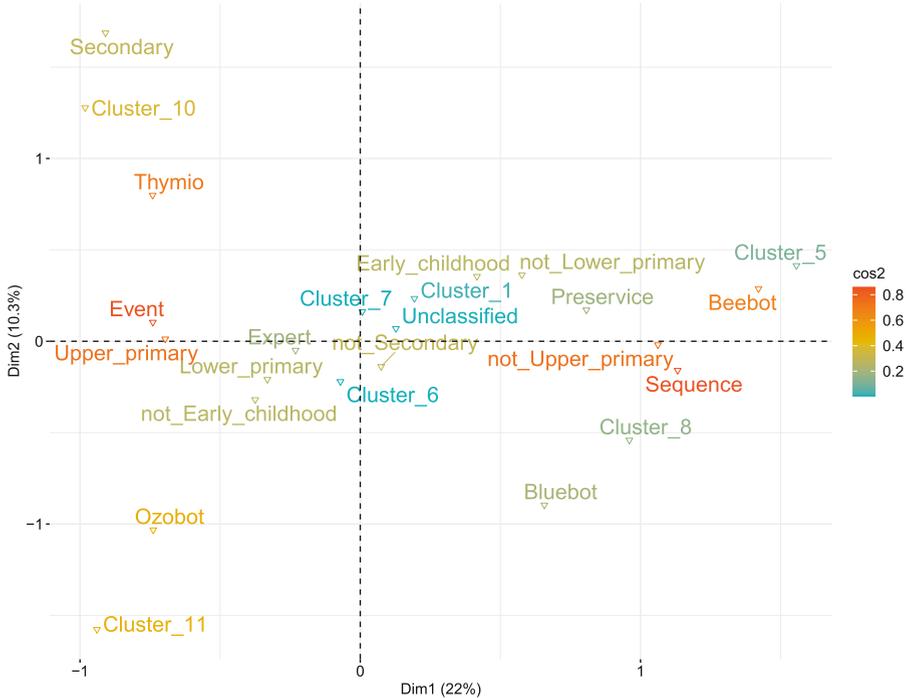

Fig. 4. Multiple Correspondence Analysis (MCA) showing clusters and variables related to documents, axes 1 and 2. Image produced in R with EXPLOR package.

4.3 Discussion

Looking at the results from Reinert's clustering, we observe that out of the seven main clusters created, five clusters are oriented toward CS content for primary and secondary (5, 6, 8, 10 and 11) and contain lexicon about movements on a grid, following paths, programming, events and machine; one cluster is related to general goals of CS teaching (1) and one is oriented toward pedagogical practices (7), with lexicon that has to do with collaborative learning and sharing knowledge with the whole class.

This let us answer RQ1: *What are the main themes that emerge from the textual analysis of educational resources for CS?* Performing clustering of our corpus, we can extract seven main clusters or topics. According to the PCK

model of Magnusson, five clusters (5, 6, 8, 10 and 11) are related to *knowledge of the curriculum*, one (1) is related to *orientations toward CS teaching* and one (7) is related to *knowledge of instructional strategies.*

We also have elements to answer RQ2, *What kind of relation can be established between the lexicon and the type of robot used in educational resources for CS?* We were able to identify the lexicon related to each of the robot present in our corpus: four clusters are significantly linked to one of the robots, since they contain the name of the robot among their overrepresented words. Cluster 5—*Manipulation* is related to *Beebot* and contains words having to do with the movement and first manipulation of an object, which could come close to direct manipulation or direct control according to Kalaš [7]; cluster 8—*Programming* is connected with *Bluebot*, with words rather related to programming notions and activities described in a way that could point to a computational control of an object [7], so in some way a more advanced phase in the process of learning CS compared to cluster 5; cluster 10—*Events* is associated with *Thymio* and contains words rather related to machines, events and behavior; and finally cluster 11—*Dancing* is associated with *Ozobot*, with words rather related to the idea of a dance. As vocabulary conveys ideas and notions, then our analysis says something about the notions used by the authors of our resources in relation with each robot.

Maybe these associations between robots and topics can appear as trivial, especially for the people who have some experience with them. But we think that it's interesting to consider that this classification did not emerge from a pre-established conceptual framework that we would have applied to the corpus; rather it emerged completely inductively from the data itself, just from a simple Textual Data Analysis where we *count words.*

Further on, this technique has revealed a difference between Beebot and Bluebot that we would not have thought about. Those robots are very similar in terms of affordance (Bluebot seems to be a kind of updated version of Beebot). However, in our resources, we see that the vocabulary associated with the two robots is somehow different, which shows that teachers connected different notions to the activities they presented for Beebot and Bluebot. Segments of text related to Beebot describe rather the first manipulations of a robot and those related to Bluebot a more advanced level of programming.

Then, MCA on Fig. 4 offers an interesting view on our data as it represents the attractions and oppositions between our variables: clusters, robots, programming paradigms, teachers' expertise and grade levels. The first observation we make is the proximity between the four clusters and the four robots to which they were associated through Reinert's clustering: cluster 10 is close to Thymio, cluster 5 is close to Beebot, cluster 11 is in the vicinity of Ozobot and cluster 8 is not far from Bluebot. We see it as a confirmation that there exists an association between these four clusters, the notions they convey, and the robots to which they are associated.

To answer RQ3, *Based on the analysis of the lexicon, what other relations can be seen between the clusters, grade levels, programming paradigms and teachers'*

expertise?, we can interpret the meaning of the axes of the MCA. We see on the first axis an opposition between *event-driven robots* on the left (Thymio, Ozobot) and *sequential robots* on the right (Beebot, Bluebot). This is consistent with the results that we had in our previous research and it shows that, considering the entire lexicon of our resources, there is an opposition in terms of vocabulary used by teachers between resources with activities on event-based robots and sequential robots. The second axis is a bit more difficult to interpret, as it opposes on one side Thymio, its cluster 10, secondary, Beebot and early childhood to Ozobot, its cluster 11, lower primary, Bluebot and its cluster 8. So there seems to be an opposition between resources for secondary or early childhood (Thymio could be for both, Beebot certainly for early childhood) and resources for primary (Ozobot and Bluebot).

It's interesting to note that teachers' expertise doesn't stand out in the MCA, as its values are situated close to the center of the plan. As we tend to consider the PCK of novices to be quite different from the PCK of more experienced teachers [9], it's a bit surprising to see that it's not a factor of differentiation in our corpus. We also note that the clusters not clearly associated with a robot (1, 6 and 7) do not play a distinctive role in the MCA.

To conclude this discussion, we think that applying computational methods to the analysis of textual data in the field of CS education is a valuable approach, because it lets us go through an important quantity of texts that would be time consuming to manage manually. Reinert's clustering helps us understand, through the clusters it produces, what topics are constituting the mental universe of teachers. It is inductive and these topics are extracted from the data itself, without having a predefined model for the interpretation. Associated with Multiple Correspondence Analysis, it gives the possibility to interpret the relation between specific variables on the lexicon and the clusters produced.

The knowledge of teachers is not an easy thing to observe: as we cannot reach it directly, we can only observe its manifestation in certain contexts or objects. Pedagogical resources produced by teachers are one of them. They constitute a trace of their planning activity and, as such, of their enacted PCK. Performing a textual analysis of the content of such resources gives access to the individual words that make them up. As knowledge is related to notions, the words that appear through textual analysis of resources say something about the knowledge of their authors.

5 Conclusion

In this research, we apply Reinert's clustering and Multiple Correspondence Analysis (MCA) to the text of pedagogical resources in the field of educational robotics considered as traces of the enacted PCK of teachers. Through the analysis of the resulting clusters and the words significantly related to them, we are able to show the main topics that teachers write about when they create resources. Here is the way we can answer our three research questions.

For RQ1 about the main themes that emerge from the textual analysis of our educational resources, Reinert's clustering creates seven main clusters representing five topics that we would classify as *knowledge of the curriculum* according to Magnusson's model of PCK, one topic related with *orientations toward CS teaching* and one topic that would be part of the *knowledge of instructional strategies*.

Regarding RQ2 about the kind of relation that can be established between the lexicon and the type of robot used in the resources, Reinert's clustering and MCA let us characterize it and say that Beebot is associated with the first manipulation of an object. Bluebot is associated with programming. Thymio is associated with events and behavior. Ozobot is associated with a dance.

Concerning RQ3 about the other relations that can be seen between the clusters, grade levels, programming paradigms and teachers' expertise, MCA shows an opposition between event-driven and sequential robots.

As a limitation for our research, we only investigated one domain of PCK: enacted PCK, and only through the analysis of the trace of teachers' planning activity. For a broader view on teachers' PCK, it would be interesting to investigate the whole pedagogical cycle (planning – enactment – reflection) and the relations that can appear between personal PCK (pPCK) and enacted PCK (ePCK) in these contexts [6].

For this reason, we wish to extend our research, using the methods of Textual Data Analysis, to other kinds of context as teachers' interviews and classroom transcripts.

References

1. Aguilar, J., Salazar, C., Velasco, H., Monsalve-Pulido, J., Montoya, E.: Comparison and evaluation of different methods for the feature extraction from educational contents. Computation **8**(2), 30 (2020)
2. Carlson, J., et al.: The refined consensus model of pedagogical content knowledge in science education. In: Hume, A., Cooper, R., Borowski, A. (eds.) Repositioning Pedagogical Content Knowledge in Teachers' Knowledge for Teaching Science, pp. 77–94. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-5898-2_2
3. Chen, X., Zou, D., Xie, H., Cheng, G.: What are MOOCs learners' concerns? text analysis of reviews for computer science courses. In: Nah, Y., Kim, C., Kim, S.H., Moon, Y.-S., Whang, S.E. (eds.) DASFAA 2020. LNCS, vol. 12115, pp. 73–79. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59413-8_6
4. Drot-Delange, B., Parriaux, G., Reffay, C.: Futurs enseignants de l'école primaire: connaissances des stratégies d'enseignement, curriculaires et disciplinaires pour l'enseignement de la programmation. RDST. Recherches en didactique des sciences et des technologies (23), 55–76 (2021)
5. Emprin, F.: Les apports d'une analyse statistique des données textuelles pour les recherches en didactique : l'exemple de la méthode Reinert. Annales de Didactique et de Sciences Cognitives. Revue internationale de didactique des mathématiques (23), 179–200 (2018)
6. Henze, I., Barendsen, E.: Unravelling student science teachers' pPCK development and the influence of personal factors using authentic data sources. In: Hume, A.,

Cooper, R., Borowski, A. (eds.) Repositioning Pedagogical Content Knowledge in Teachers' Knowledge for Teaching Science, pp. 203–223. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-5898-2_9

7. Kalaš, I.: Programming in lower primary years: design principles and powerful ideas. In: Dagienė, V., Jasutė, E. (eds.) Constructionism 2018 – Constructionism, Computational Thinking and Educational Innovation: Conference Proceedings (2018)

8. Wessel, L.: Vocabulary in learning processes towards conceptual understanding of equivalent fractions–specifying students' language demands on the basis of lexical trace analyses. Math. Educ. Res. J. **32**(4), 653–681 (2020)

9. Magnusson, S., Krajcik, J., Borko, H.: Nature, sources, and development of pedagogical content knowledge for science teaching. In: Gess-Newsome, J., Lederman, N.G. (eds.) Examining Pedagogical Content Knowledge, vol. 6, pp. 95–132. Springer, Dordrecht (1999). https://doi.org/10.1007/0-306-47217-1_4

10. Mercer, N.: The analysis of classroom talk: methods and methodologies. Br. J. Educ. Psychol. **80**(1), 1–14 (2010). https://onlinelibrary.wiley.com/doi/pdf/10.1348/000709909X479853

11. Prieur, M., Monod-Ansaldi, R., Fontanieu, V.: Réception des démarches d'investigation prescrites par les enseignants de sciences et de technologie. RDST. Recherches en didactique des sciences et des technologies (7) (2013)

12. R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2022)

13. Reffay, C., Parriaux, G., Drot-Delange, B., Khaneboubi, M.: Robotics in primary education: a lexical analysis of teachers' resources across robots. In Keane, T., Lewin, C., Brinda, T., Bottino, R., (eds.) Towards a Collaborative Society Through Creative Learning: IFIP World Conference on Computers in Education, WCCE 2022, Hiroshima, Japan, 20–24 August 2022, Revised Selected Papers. Springer International Publishing (2023). https://link.springer.com/book/9783031433924

14. Reinert, A.: Une méthode de classification descendante hiérarchique: application á l'analyse lexicale par contexte. Cahiers de l'Analyse des Données **8**(2), 187–198 (1983)

15. Sherin, B.: A computational study of commonsense science: an exploration in the automated analysis of clinical interview data. J. Learn. Sci. **22**(4), 600–638 (2013)

16. Shulman, L.S.: Those who understand: knowledge growth in teaching. Educ. Res. **15**(2), 4–14 (1986)

17. Silva, D.B., Silla, C.N.: Evaluation of students programming skills on a computer programming course with a hierarchical clustering algorithm. In: 2020 IEEE Frontiers in Education Conference (FIE), pp. 1–9. IEEE, Uppsala, Sweden (2020)

18. Sobczak, A., Debucquet, G., Havard, C.: The impact of higher education on students' and young managers' perception of companies and CSR: an exploratory analysis. Corp. Governance: Int. J. Bus. Soc. **6**(4), 463–474 (2006)

Reshaping Unplugged Computer Science Workshops for Primary School Education

Martina Landman[✉] , Sophie Rain , Laura Kovács ,
and Gerald Futschek

TU Wien, Vienna, Austria
{martina.landman,sophie.rain,laura.kovacs,
gerald.futschek}@tuwien.ac.at

Abstract. Through meticulous analysis and adaptation, we reshaped unplugged computer science activities to align with the developmental needs and capabilities of primary school children. Our approach focuses on distilling the essence of computer science topics while tailoring their content and delivery methods to suit the younger audience. We describe our efforts and report on our experiences implementing our framework for eight primary school classes, turning our unplugged computer science workshops for secondary school classes into an educational playground for 192 primary school children. Our work contributes to the general societal mission of supporting more and more children to become interested in STEM, ensuring that our technological future is as diverse as possible.

Keywords: Primary Education · CS Unplugged · Girls Empowerment

1 Introduction

In today's highly autonomous world, Computer Science (CS) has become a subject everyone should learn about. Within the work presented in this paper, we advocate that computer science should be accessible to everyone, starting with our society's youngest generation. We therefore designed an unplugged CS workshop framework to ease the integration of CS topics within primary school education. We focus on adjusting a set of CS unplugged activities [1] adapted by us, initially designed for an informal learning environment of a 90 min workshop targeting early secondary school students, into engaging activities suitable for primary school students.

   With the aim of reshaping unplugged CS activities, we carefully selected three CS topics, forming the backbone of our unplugged CS workshops for primary schools. Through the engaging scenarios of (Task 1) sorting, (Task 2) searching, and (Task 3) planning, we provide young children the opportunity to explore and experiment with various CS concepts playfully. This way, we empower primary school children to create their own solutions (i.e. algorithms) without compromising the conveyed CS content. In particular, we address (Task 1) the role of fast-moving (secret) data points within sorting algorithms; (Task 2) the concept

of divide and conquer algorithms during searching/playing cards; and (Task 3) the need for optimal planning while solving path-finding puzzles, allowing us to ultimately expose children to basic ideas of programming (algorithms, programming languages).

To make our workshop tasks as accessible as possible, especially for first-graders, our CS activities are developed without requiring reading or writing skills (Sect. 4). Further, we carefully integrated age-appropriate content with CS tasks (Sect. 5) and paid special attention to the children's social-emotional development during our unplugged CS tasks (Sect. 6).

We performed eight iterations of our unplugged CS workshops for primary schools. In each such iteration, one primary school class of 22–27 students (ages 8–9) visited our workshop at our institute, as this gives the children the opportunity to experience a scientific institution. In total, 192 primary school children have participated in our CS workshops so far (Sect. 6). Based on our empirical evaluation and the teachers'/students' feedback, we conclude that reshaping CS workshop tasks for primary schools is a worthy and successful effort, which we aim to further strengthen at TU Wien.

2   Background and Related Work

2.1   In-House Outreach Programmes

At TU Wien, we have already conducted outreach programmes for teaching CS concepts to secondary school students. Within this existing framework, we offer (i) online programming courses to ease individual learning; (ii) in-class programming courses at interested schools, and (iii) "unplugged" CS workshops at our university. Each event from (i)-(iii) is conducted by the authors, based on our experience as university students/assistants/professors. The work described in this paper carefully reshapes item (iii) and proposes "unplugged" CS workshops as a supplementary educational programme for primary schools.

2.2   Related Approaches and Initiatives

A recent report by the Brookings Institution [9] points out the need for computing education "early on", that is, as early as primary and secondary schools, and identifies the key challenges for doing so. In her blog, Sue Sentance [7] highlights the main lessons learned from [9], importantly to "start early" with CS education. While [9] finds that there is a worldwide interest in the early promotion of CS programmes, it also stresses the need to catch up with up-to-date CS developments and improve the overall integration of CS in various curricula.

In addition to coding and programming initiatives[1], there are creative approaches to teaching CS without a computer, most notably CS Unplugged [1], Abenteuer Informatik [5] and Bebras Challenge[2]. Our own outreach programs are mainly based on tasks inspired by the former two programs. We also

---

[1] e.g. https://code.org/.
[2] e.g. https://bebras.org/.

extend these initiatives with further ideas, such as including tasks involving simple robotics. In particular, we use Ozobots[3] as our main robotics device because of their ability to be programmed through colour codes. Ozobots are thus immediately accessible to students of all ages and allow their programming via "paper and pencil", which fits nicely with our unplugged CS activities.

It has been proposed that curricula and long-term initiatives are most effective when adhering to the concept of a spiral curriculum [3]. Extending and revising our CS tasks to younger students, as proposed within our current paper, should thereby ideally offer an iteration of our existing program for a younger age group, allowing them to be exposed repeatedly to the same core ideas, each time in a format that is most appropriate for their respective age and stage. Two of the most significant advantages of such an approach are increased retention through repetition, as well as the possibility to continually increase task complexity and therefore reach technical depth.

Instructional design models support structured and traceable developments of teaching materials and tasks. Among others, the ADDIE model [2] implements such a structured process within 5 main phases "analysis", "design", "development", "implementation" and "evaluation".

To ensure high-quality (unplugged) CS workshops, design-based research (DBR) [6] can be used, characterized by the cyclical repetition of a learning intervention to improve it. The Action Research Model [8], consisting again of cyclical, repetitive phases in which the teacher herself reflects on and evaluates her action, is particularly suitable. There are different interpretations of the phases, such as "plan", "act", "observe" and "reflect" [4], which provide a framework for collecting data from one's own experience of delivering a learning intervention and building on this to improve.

3 Methodology

In this section we present the workflow for reshaping our existing unplugged CS workshop for secondary schools, allowing us to introduce unplugged CS workshops for primary school children (ages 6–10, school grades 1–4). Within this workflow, we systematically answer the following research question:

> **RQ:** How can secondary school tasks be simplified in an informal learning setting (workshop) to be used in primary school level?

We address this RQ by analyzing the difficulties and needs of the target group; revising and designing new learning tasks; and evaluating and improving our learning tasks based on first-hand experience collected within eight workshops implementation of RQ, allowing us to attract 192 primary school children (ages 8–9). Doing so, we rely upon the ADDIE instructional design model [2]. We restructure learning materials from secondary schools and integrate our solutions

---

[3] e.g. https://ozobot.com/.

to the RQ within the five main ADDIE phases, as summarized in Fig. 1 and discussed next.
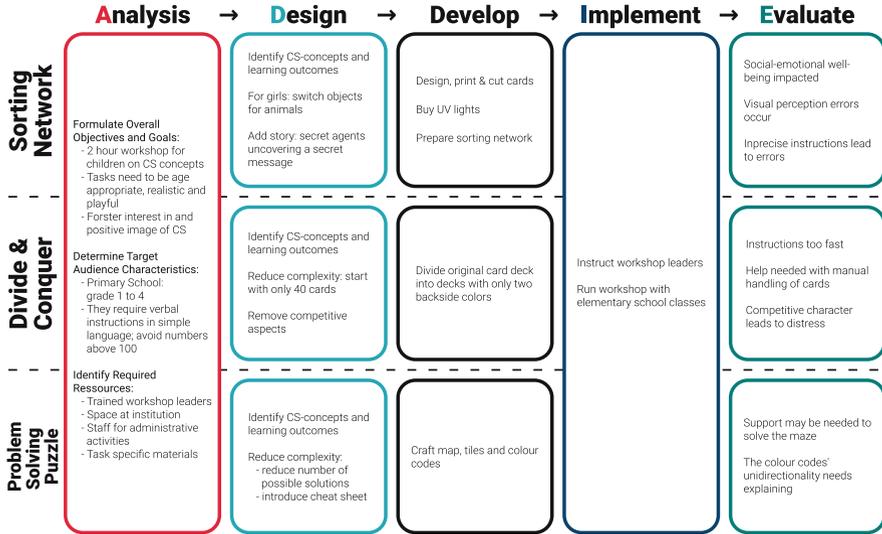

Fig. 1. Summary of the main steps within each ADDIE phase of our approach.

Analysis. Through close observation of the previous practical implementation of our CS tasks for secondary school classes, we identified learning difficulties. We analyzed these challenges by focusing on our new target group of primary school children (Sect. 4).

Design. We identified and revised CS tasks to be presented in our unplugged CS workshops. Each CS task is motivated and related towards complementing primary school education with CS topics. Since we aim for a diverse environment, we considered the findings of [10] in the design of our CS tasks. For example, as [10] argues that girls are more interested in a problem framed about humans/animals instead of using objects, whereas boys are indifferent to such problem-framing choices. We therefore designed our CS tasks using visual material involving animals, thus favouring girls-friendly content while not harming boys' perceptions (Sect. 5). The conveyed CS topics within our workshops are abstracted on a high level to make the concepts accessible to primary school children.

Development. We formalized the learning objectives for each CS task and created a prototype of each CS unplugged task. We make this prototype accessible also to future workshop leaders (Sect. 5).

Implementation. We instruct team members and university students to serve as workshop leaders and promote the workshops to schools.

Evaluation. After each workshop, we conduct a joint group reflection inspired by the reflection phase of action research (Sect. 6). This information is used to continuously improve our CS tasks and our unplugged CS workshop setting.

4 Task Analysis and Evolution

We now detail the evolution of our tasks, originally introduced for secondary schools and reshaped for primary schools (see Sect. 5). We provide the rationale for our changes while relying on the ADDIE model [2].

Within our first ADDIE phase of "Analysis", the following three main points have been considered: analyzing the characteristics of our target group, formalizing our learning objectives, and identifying required resources for implementation. With the aim of reshaping existing unplugged CS workshops from the secondary level to the primary school level, we set the following aspects.

Overall Goal. We offer unplugged CS workshops, with each workshop instance taking place at our university for a duration of two hours. Our overall goal is for young children in primary schools to learn core computer science concepts. Looking through the possible tasks and concepts of our existing unplugged CS workshops for secondary schools, we decided on choosing the following three workshop tasks for primary schools: *(Task 1)* using sorting networks; *(Task 2)* sort and search via the CS principle of Divide and Conquer concept; and *(Task 3)* planning via solving puzzles based on computational thinking and experimenting using Ozobots. We believe these tasks can be made age-appropriate, realistic, and playful (see Sect. 5), sparkling genuine interest in computer science at a very young age. We also paid special attention to making these tasks as diverse as possible, in particular friendly to young girls.

Target Audience Characteristics. Our workshops target primary school children of any age without any prior knowledge. We therefore provide easy-to-adjust workshop tasks, for example, by using small number ranges of 1–9 for first grade children vs larger number ranges of 1–100 for third graders in Task 1. Similarly, we initially use card stacks of 4×10 cards in Task 2, instead of starting immediately with all $4 \times 4 \times 10$ cards. Whereas in Task 3, we reduced the possibilities of different paths in the map added additional tiles for path planning and introduced a "cheat" sheet that children could use to identify colour codes.

Similarly, reading comprehension should not be a hard requirement in our primary school workshops. We used visual material, such as different-sized animals for sorting in Task 1 instead. To make our tasks more playful for primary school children, we use secret messages and UV light-pens in Task 1 and obstacles as animals/ponds/forests in Task 3.

Finally, instead of using written instructions (as is the case for secondary schools), we verbally explain our workshop tasks to primary school children, using simple, age-appropriate terms.

Resources. To run our workshops at TU Wien, skilled workshop leaders are needed. Therefore, we dedicated ourselves to training computer science undergraduate/graduate students, helping us as workshop leaders. This way the school

teachers could focus on the children's social-emotional well-being. We also implemented adequate framework conditions for our workshops, including reserving suitable rooms; establishing and maintaining contact with school classes; and organizing task-specific material and equipment.

5   Task Design and Development

Within the design of our CS workshops and their respective tasks, we placed special emphasis on creating a suitable learning environment for young children, where the framework focuses on embedding the learning content within an age-appropriate scene. It is essential to assist children in adjusting to their new surroundings; we therefore relate our university setting to the daily school life scenario of students, while drawing similarities between primary school and university activities.

For each CS task, we considered the following design process. First, we **(M)** motivate the task by explaining why the task is relevant to daily life. Second, we let children **(E)** experiment with the task, develop different solutions, and implement their own strategies (i.e. algorithms). Finally, we **(R)** relate the task to CS topics by providing them answers on how such CS topics (and solutions) are used on a daily basis. In conclusion of this design process, we developed an effective task prototype and set the **learning outcomes** of our workshop tasks.

5.1   Task 1 – Sorting Network

(M) Motivate. This task is inspired by the unplugged CS task of sorting networks[4], enabling children to sort items by following a small number of instructions. We motivate the use of sorting networks, and in general sorting, with real-life sorting scenarios of clothes (by size), videos (by length), and groceries (by price).

(E) Experiment. We use two large sorting networks taped to the floor, supporting six children to sort items according to some predefined property. Six children *start* sorting by one child standing in one of the starting nodes (represented as squares) and randomly choosing, for example, one number card from a collection (numbers 1–100[5]). The chosen cards are unsorted, with the cards becoming sorted upon the children *finish* traversing the sorting network and arriving on the other side of the room in correctly sorted order in the designated boxes on the floor.

For traversing the sorting network, the children *initialize* the sorting process by moving from their initial position along lines, arriving at the first node of the network, which is represented as an ellipse on the floor where two children fit in.

---

[4] https://www.csunplugged.org/en/topics/sorting-networks/reinforcing-numeracy-through-a-sorting-network-junior/.

[5] For younger children, we use numbers 1–9 or simple geometric figures (e.g. triangles) of different sizes.

Within the ellipse node, *decisions* between two children are made by comparing the values of the property of interest: for example, when using number cards, the child with the smaller (resp. bigger) number moves along a red (resp. blue) line into the next ellipse node. Such decisions between pairs of children are *repeated until* each child reaches their (final) spot, resulting in a sequence of sorted numbers. In such a sorting process, children thus implement four instructions (start, initialize, decide&repeat, stop) within the sorting network; moreover, children are encouraged to experiment with "runs" of the sorting network, trying to become faster while correctly executing the sorting instructions.

We further experiment with sorting secret cards, each card hides a secret key that is only visible when a UV light shines on the cards. Here, we used animals in different sizes on each card. The children sort secret cards as before, yet, to make "decisions" in the black ellipse nodes, the children need to wait for a UV light pen to process and compare the secret keys. While experimenting with this, children realize that more UV light pens significantly speed up the sorting process by parallelizing sorting instead of waiting for one single central pen (processor).

(R) Relate. Children are introduced to sorting networks and algorithms, enabling them to sort items parallel, similarly to sorting engines on the web operate. Within this process, children learn about (fast) algorithms, allowing our workshop task to explain foundation concepts from the CS areas of Algorithms and Data Structures, as well as Parallel Computing.

Learning Outcomes. As the underlying CS concepts of this task are parallelization and rigorous algorithmic execution, we set the following learning objectives, as learners can:

– understand that the more people make comparison decisions simultaneously, the faster the sorting network/algorithm will (correctly) terminate;
– solve the algorithmic aspect of sorting networks by following the given instructions on traversing sorting networks.

5.2 Task 2 – Divide and Conquer

(M) Motivate. During this task, we motivate the CS principle of Divide and Conquer by instrumenting children to playfully develop their own set of instructions that yield sorting algorithms for a shuffled stack of cards. We exemplify the need for such sorting algorithms for finding items in large collections, such as identifying that a particular book with 2–3 characteristics is missing in large library collections.

(E) Experiment. We split children into groups of five–seven participants, urging collaborative teamwork. We use Ligretto cards[6] as task workhorses since these cards exhibit easily visible, three different properties: each card has one of four colours on the card back and a number between 1–10 on its front side. A full set of Ligretto cards thus consists of $4 \times 4 \times 10 = 160$ cards.

---

[6] https://www.schmidtspiele.de/detail/product/ligretto-blue.html.

Each group of children receives only one shuffled stack of Ligretto cards[7], with one side of the cards being of the same (front) colour. The task is to sort these cards based on (back) colour and numbers as quickly as possible. After visually illustrating the desired sorting outcome (e.g. red cards from 1– 10, followed by blue cards from 1–10, etc.), we give no further instructions on how sorting should be achieved; yet, if children struggle with structuring their ideas, further hints may be given by workshop leaders. We encourage children to discuss their ideas together and have everyone's role determined before the sorting process starts. After the groups complete sorting their shuffled stack of cards, each group's solution is discussed, with the ultimate goal that groups learn from other groups' solutions. Within this discussion, we highlight the key steps children used in their solutions: e.g. divided the big stack of cards into smaller ones, sort the smaller stacks, and then merge sorted card stacks; in other words, children intuitively implemented the CS principle of Divide and Conquer.

A second round of card sorting is started, without changing groups, but only shuffling cards. The revised sorting solutions are again discussed, again focusing on whether sorting performances have improved compared to the previous round. In case of significant improvements, e.g. sorting times fit within a short time window (1–2 min), another round of card sorting is initiated with increased difficulty: a second set of 4×10 shuffled cards of another front-side colour are additionally used, asking children to sort based on two colours and numbers 1– 10. Children are likely to adapt their previous solutions to fit the new challenge. Nevertheless, Divide and Conquer remained the crux of their solutions.

Finally, we also initiate a card sorting process where workshop leaders silently remove one card from a complete stack. When asking children to sort the shuffled stack of cards, children identify in a relatively short amount of time that one card is missing and precisely characterize the colour and number of the missing card. However, again, all this while playfully executing their own solutions based on Divide and Conquer.

(R) Relate. When sorting large data sets or searching for an item in a big database (i.e. Ligretto cards), children naturally work collaboratively in order to find their fastest solutions: divide a given problem into subproblems and merge the sorted subproblems into the sorted version of the given problem. Such a structured approach is the basis of Divide and Conquer, and hence children are naturally introduced to key concepts within the CS areas of Distributed Computing and Resource Optimization.

Learning Outcomes. While parallelization is a practical application of Divide and Conquer, its key benefit comes from dividing an intractable problem into more manageable parts. The learning outcomes of this task are therefore that learners can:

– identify/recognize subtasks;
– divide a task into subtasks and distribute the solving workload among group participants;

---

[7] we only use 4×10 cards per group initially.

– describe a (Divide and Conquer) strategy to sort a shuffled stack of cards.

5.3   Task 3 – Planning in Puzzle Solving

(M) Motivate. During this task, children create (optimal) paths within a (geo-graphic) map given limited resources. We motivate the need for planning via route finding from school to home, or in general for other routes, frequently used by automotive navigation systems. To showcase the impact of human/children intelligence vs computer intelligence, students also discover how to steer a small line-follower robot using colour codes within the path created by the children.

(E) Experiment. We split the whole group into groups of three–five partic-ipants. Each group receives an A2 sheet with a printed tile grid, as shown in Fig. 2(a). Each printed tile is either empty, contains a picture of an object, or has a partial path, i.e. a black line drawn from one side to another. The pictures used within the tile include a school, a house (home), and the obstacles of a forest, a pond, and a tiger. Children are further given a limited set of additional path tiles with either a straight or a curved path. Each group starts with finding a path from the school to the house while avoiding the a priori given obstacles. The children may use only the given set of straight/curved path tiles, as shown in Fig. 2(b). Once the students have found a suitable path, they are given an Ozobot placed at the school. This Ozobot will follow the developed path. Hence, children already know what the Ozobot should do: do exactly what the children's solutions tell them to do and thus reach the house (home).

In a further step, a new tile grid with intersections is introduced. When encountering an intersection, an Ozobot may randomly choose which direction to take. To have full control over the Ozobot, children are encouraged to instru-ment the Ozobot by laying down a colour code (over the path) right before the intersection, determining the specific direction to be taken. In other words, children "program" the Ozobot to follow their chosen path.

Once the children find solutions, the workshop instructors visualize possible solutions with live coding using the block-programming language "Scratch"[8]. Children are thus introduced to the art of programming only after they know what a computer scientist should do: understand and create a path-finding solu-tion (algorithm) just like the children did.

(R) Relate. When planning a path from home to school, children ensure that their solutions are safe, e.g. obey a priori given rules and omit forbidden/unsecure items in their planned trajectory. As such, the children are playfully introduced to the CS areas of Formal Methods and Computer Security, and in general to Artificial Intelligence.

Learning Outcomes. This task corresponds to solving a logical puzzle: find-ing a way through a maze/map with a limited range of tiles. The possibilities are limited and only a few ways are possible, thus connecting naturally to back-tracking concepts. This task also conveys elementary knowledge about computer

---

[8] https://scratch.mit.edu/.
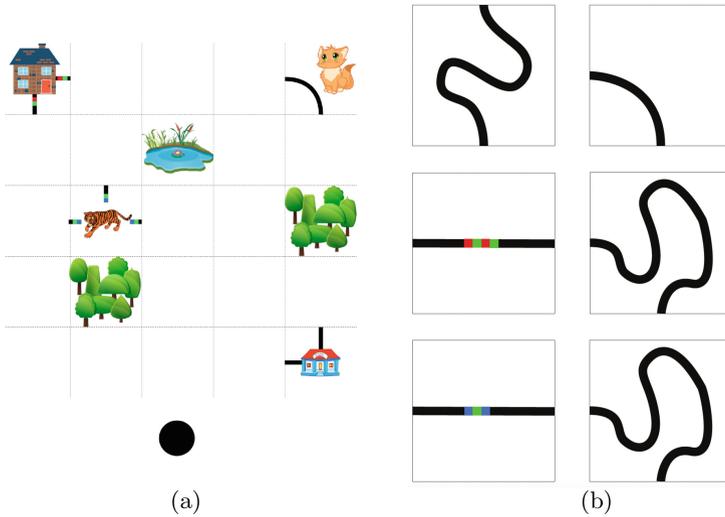
Fig. 2. (a) Finding a path from school to home, (b) using predefined tiles.

commands and languages that computer systems and machines can understand. We identified the following learning outcomes as learners can:

– plan a route given additional constraints;
– describe different possible solutions by using less/more constraints;
– understand that computer programs/machines need commands in a specific programming language (e.g. colour codes for Ozobots).

6   Task Implementation and Evaluation

For the implementation, the executing student employees were well-instructed. There were written instructions, a clear distribution of tasks, and a joint preliminary discussion. It was clearly defined in advance who is responsible for which activities during the workshop, how the tasks work, and which things they must pay special attention to when working with children.

In the final "Evaluation" phase of the ADDIE model, we conducted an action research-inspired approach through eight pilot runs of our unplugged CS workshops for primary schools with reflection sessions after each run. Altogether, we reached 192 primary school students. Each workshop instance was evaluated through the cyclical phases of "plan", "act", "observe", and "reflect" [4], allowing us to revise further and improve workshop materials and the overall workshop process, as detailed below.

Evaluation of Task 1 – Sorting Network. Based on the evaluation of our CS workshops so far, we made changes in Task 1 targeting the followings:

- *Social-emotional well-being:* As there are only 6 input nodes in the sorting network , only 6 students can experiment per network. Since class sizes may vary, with our original setup (using two sorted networks) not all students got to experiment the same number of times. Originally, only three runs of the sorting network were planned (first run with numbers, second run with invisible animal images and one UV Light "comparator", third run with three "comparators"). This initial plan resulted in having a few children that experimented with sorting networks only once, while other children twice. Due to the children's emotional reactions, we extended Task 1 with a further run with numbers so that each child tries out our sorting network at least twice. Empty inputs nodes/squares are filled with teachers/volunteering children.

- *Visual perception error:* One observed difficulty students had was reading similar numbers (e.g. 6 and 9). In the joint reflection sessions of our workshops, this topic came up several times and we kept sorting out numbers that could be misunderstood; for example, eliminating 6, 9, 66, and 99.
- *Instructional error:* We observed that children might tend to go in the wrong direction of the sorting network: though the instruction is "follow the blue line if you have the bigger number/animal", there is also a blue line leading back to the start . In particular, we observed chaotic behaviour when the starting phase of sorting was overcrowded. In our workshop reflection rounds, we agreed to pay more attention to the fact that the children go to the starting squares; we also included this as a (start) instruction for our task.

Evaluation of Task 2 – Divide and Conquer. Similarly to the reflection rounds and observations made withing Task 1, the following changes have been made on Task 2 to address:

- *Instructional error:* We spend more time explaining the task while providing an a priori fixed set of hints on-demand.
- *Social-emotional well-being:* We playfully motivate competitive efforts while measuring the times children spent on sorting cards, emphasizing that groups do not compete against each other but essentially only improve their own times (due to different solutions they use).

Evaluation of Task 3 – Planning in Puzzle Solving. Similarly to Tasks 1–2, the following changes on Task 3 have been implemented to prevent:

- *Procedural errors:* We actively interact with children and provide hints for using the tiles/planning the path.
- *Conceptual misunderstandings:* Our Ozobots need to read (i.e. drive over) a colour code for one specific direction (left/right/straight). The tiles indicate such directions by arrows next to the colour code. For example, a tile with a blue-black-red sticker is the command "go straight ahead"; yet, if the Ozobot reads this code backward, it chooses a random direction. As the sequence of colors in our tiles are not very self-explanatory, we allocate extra time so children experiment more with coloured tiles.

7   Conclusions

We showed how to reshape unplugged CS tasks from secondary to primary schools, providing an age-appropriate and diverse CS workshop environment. We rely on the design-based research approach advocated by the ADDIE structured workflow model and continuously adjust our learning framework by reflecting on our CS workshop experiences and performances.

Through our unplugged CS workshops reshaped for primary schools, we motivate and relate CS topics from and to daily life and encourage children to experiment with their own solutions. Based on our task evaluation so far, we believe that unplugged CS activities can (and should) nicely complement primary school education. We aim for further developments of our CS task by exploring the participant's background and adjusting workshop instructions/algorithmic solving challenges accordingly.

Appendix

Sorting Network
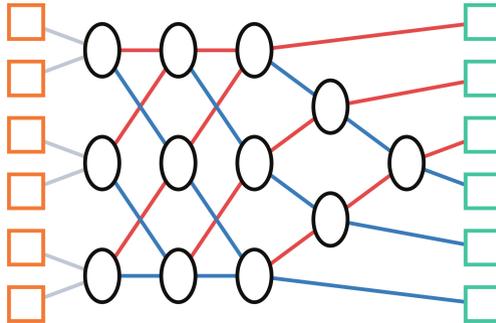
Fig. 3. Sorting network used in task 1 with input/output (orange/green squares) from the CS unplugged task

References

1. Bell, T., Witten, I., Fellows, M.: CS Unplugged: an enrichment and extension programme for primary-aged students (2015). https://www.csunplugged.org/
2. Branch, R.M.: Instructional Design: The ADDIE Approach, 1st edn. Springer, Boston (2009). https://doi.org/10.1007/978-0-387-09506-6
3. Bruner, J.S.: The Process of Education. Harvard University Press, Cambridge (2009)
4. Dickens, L., Watkins, K.: Action research: rethinking lewin. Manag. Learn. **30**(2), 127–140 (1999). https://doi.org/10.1177/1350507699302002
5. Gallenbacher, J.: Abenteuer Informatik (2008). https://www.abenteuer-informatik.de/
6. Easterday, M.W., Lewis, D.R., Gerber, E.M.: Design-based research process: problems, phases, and applications. In: Proceedings of International Conference of the Learning Sciences, ICLS 1(January), pp. 317–324 (2014). https://www.scholars.northwestern.edu/en/publications/design-based-research-process-problems-phases-and-applications
7. Sentance, S.: Computer science education is a global challenge (2021). https://www.raspberrypi.org/blog/brookings-report-global-computer-science-education-policy/
8. Stephen, M.: Corey: action research in education. J. Educ. Res. **47**(5), 375–380 (1954). https://doi.org/10.1080/00220671.1954.10882121
9. Vegas, E., Hansen, M., Fowler, B.: Building skills for life: How to expand and improve computer science education around the world. Brookings (2021). https://www.brookings.edu/essay/building-skills-for-life-how-to-expand-and-improve-computer-science-education-around-the-world/
10. Vorvoreanu, M., Zhang, L., Huang, Y.H., Hilderbrand, C., Steine-Hanson, Z., Burnett, M.: From gender biases to gender-inclusive design: an empirical investigation. In: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. CHI 2019, New York, NY, USA, pp. 1–14. Association for Computing Machinery (2019). https://doi.org/10.1145/3290605.3300283

Curricula and Computer Science Concepts

Evaluating the New Secondary Informatics Curriculum in The Netherlands
The Teachers' Perspective

Nataša Grgurina[1,2]([✉]) [iD], Jos Tolboom[1] [iD], and Bart Penning de Vries[1]

[1] SLO, Amersfoort, The Netherlands
n.grgurina@rug.nl, {j.tolboom,b.penningdevries}@slo.nl
[2] University of Groningen, Groningen, The Netherlands

Abstract. Since the introduction of Informatics as an elective course in secondary education in the Netherlands in 1998, the implemented curriculum is being regularly monitored. The results of the large 2013 secondary Informatics teachers survey contributed to the revision of the Informatics curriculum. This revised curriculum came into effect in 2019. In line with regular curriculum monitoring practices, the Netherlands Institute for Curriculum Development is polling the secondary Informatics teachers to understand their views and opinions on the intended curriculum and to learn about their implemented curriculum The results indicate that the majority of the respondents find the new Informatics curriculum better than the old one and that it offers a solid foundation for their teaching practice. A minority either misses some content in the curriculum or considers it overloaded with content, and some find it not to be up to date. Furthermore, the results of this survey are compared to the results of the 2013 survey to assess to what extent the new Informatics curriculum meets the teachers' needs and recommendations better.

Keywords: Informatics curriculum · Computing curriculum · Secondary education

1 The Position of Informatics in The Netherlands

In the Netherlands, the Informatics course in secondary education was introduced in 1998, during a reform of the upper secondary education. Informatics is positioned as an elective subject taught in the 10th and 11th grade of senior secondary education (in Dutch: HAVO) and grades 10 though 12 of pre-university education (in Dutch: VWO). Upper secondary education in HAVO and VWO in the Netherlands is split up in four so called profiles. Each profile serves a specific target group of the student population. These target groups are stratified based on the students' interests and the demands of their expected further education. The two humanities profiles are called Culture & Society (C&S), and Economy & Society (E&S). The two science profiles are Nature and Health (N&H), and Nature and technology (N&T). The elective subject of Informatics is to be offered in all of these four profiles. The schools, however, are not required to offer Informatics. This

implicates that the curriculum has to be appealing for all of the upper secondary school students. That is why twelve elective themes were added to the curriculum, broadly varying from 'Social and individual impact of informatics' (humanities oriented) to 'Computer architecture' (technology oriented).

Contrary to most of the other subjects, Informatics is assessed through a school exam only and not through a national exam at the end of secondary education. When it was initially introduced in 1998, the curriculum—i.e., the learning standards—consisted of 53 learning goals specified in great detail. In 2007, the curriculum was streamlined resulting in a list of 18 briefly outlined learning goals [6]. In 2013, the government commissioned an inquiry resulting in a report by the Netherlands Institute for Curriculum Development (in Dutch: Stichting Leerplanontwikkeling, SLO) to explore the teachers' ideas about the necessity to change the Informatics curriculum and their views on a possible introduction of a national examination for Informatics. Similarly to the inquiries in other countries [2, 8] this opportunity was seized to chart the characteristics of the Informatics teacher population regarding their demographics, educational background and teaching experience as well [10]. Within weeks of the publication of this report in 2014, the government appointed a committee to revise the Informatics curriculum. The gist of that Dutch report and the events leading to this new informatics curriculum are described in detail in [5] The curriculum revision process finished in 2016 and resulted in a new Informatics curriculum which met a number of set requirements, including not favoring any specific pedagogic approach and being able to count on the wide support of the teacher community. This curriculum contains 18 learning objectives. Six of them are compulsory themes forming the core curriculum: (A) Skills, (B) Foundations, (C) Information, (D) Programming, (E) Architecture, and (F) Interaction. Another twelve themes are elective: (G) Algorithms, computability and logic, (H) Databases, (I) Cognitive computing, (J) Programming paradigms, (K) Computer architecture, (L) Networks, (M) Physical computing, (N) Security, (O) Usability, (P) User Experience, (Q) Social and individual impact of informatics, and (R) Computational Science. In addition to learning about the core curriculum, the HAVO students are required to learn two elective themes and the VWO students four elective themes as well. The curriculum came into effect in 2019 [1]. By that time, the textbook publishers produced teaching materials for the core curriculum, while the teaching materials for elective themes were produced by teacher teams under guidance by SLO [5]. (In the rest of this paper, we refer to these teaching materials as SLO teaching materials.) Several parties provided professional development courses for teachers, primarily focusing on the newly introduced topics.

In 2022, the first cohort of the students attending pre-university education and a second cohort of those attending senior secondary education have finished the Informatics course based on this new curriculum. In line with the Dutch curriculum monitoring policy, this signals the moment for SLO to start to check how the intended curriculum, containing the description of learning goals and underlying rationale, works in classroom practice. The examination of other aspects of the implemented and the attained curriculum—such as looking at the actual process of teaching and learning, and at learning experiences and learning outcomes of the learners—are mostly beyond the scope of the regular monitoring process by SLO.

As our work summarized in [5] describes the intended curriculum, this paper focuses on the evaluation of the implemented curriculum [11] as it is perceived and interpreted by the teachers. Furthermore, it reports on the status of the subject in schools and the context of Informatics in a broad sense.

The research questions we strive to answer are:

1. What are the experiences of Dutch informatics teachers with the new informatics curriculum?
2. To what extent are Dutch informatics teachers satisfied with the new informatics curriculum?

2 Background and Method of the Evaluation

This study takes the Dutch informatics curriculum as described in [6] as a point of departure and reports on the next step taken in 2022–2023 in The Netherlands with respect to a systematic way of evaluating the curriculum.

In this chapter, we describe this step and the method used, and we first look back at the history of Dutch secondary informatics education so far. In its current form, as stated above, the curriculum stems from 1998 [6]. Apart from a technical curriculum alteration in 2007, it was not before 2013 that policymakers acknowledged the pressing need for reform the Informatics curriculum. By then, the Royal Netherlands Academy of Arts and Sciences (Dutch: KNAW) published a report Digital literacy in secondary education [7]. This report, among other issues, asked for a reform of the Dutch informatics curriculum and states, "At general secondary (HAVO) and pre-university (VWO) levels, the subjects Information science and Informatics have a marginal status. Their quality is insufficient and their content is outdated. Urgent action is needed." (p10). It goes on to recommend to, "Completely overhaul the optional subject Informatics in the upper years of general secondary education (HAVO) and pre-university education (VWO). By a flexible and modular design, the subject should remain up to date and appeal to students regardless of their focus area." (p11).

The Dutch Ministry of Education, Culture and Science requested SLO to investigate whether these recommendations were supported by the population of Dutch informatics teachers. In the resulting report [10], the results of a literature review, teacher interviews, a survey and expert consultations are described. The survey was completed by almost 60% of the teacher population. This survey conducted in 2013 served as the initial reference for the survey conducted in 2022. Its purpose was to examine whether the recommendations put forward by the teachers, which had also been confirmed through literature reviews and interviews with both teachers and subject matter experts, were effectively implemented in the curriculum introduced in 2019 and fulfilled their expectations.

SLO, the Netherlands Institute for Curriculum Development, has extensive experience in setting up and conducting research regarding the evaluation of the implemented curriculum, mainly carried out by its department of Advice and Research. In fact, SLO did so following every single curriculum reform that has been developed in the Netherlands for the last decades. These reforms were usually initiated by urgent need in some discipline, and there was never a systematic redesign of the secondary curriculum in full width, but only for one subject at a time, or sometimes a few of subjects simultaneously.

This led to a tailor-made evaluation practice, and this way, longitudinally monitoring has been impossible. In our view, this practice is not optimal, and therefore, by re-using the format of our 2013 Informatics curriculum evaluation, we hope to contribute to a more systematic and consistent way of curriculum evaluation. There are cautious movements in The Netherlands that the whole system of curriculum development will be set up in a more structured way, concerning all subjects, repeatedly over a constant period of time. This way, a more comparable way of evaluation will be facilitated. The fact that in this paper we report about data that have been collected with an instrument that has been used before may hopefully be seen as a first step. In 2019, Falkner and colleagues reported on an evaluation instrument called MEasuring TeacheR Enacted Computing Curriculum (METRECC) [4]. The goal of this instrument is to survey teachers in K-12 schools about their implementation of informatics curriculum to understand pedagogy, practice, resources and experiences in classrooms around the world. The reason we did not use METRECC was that we chose to be consistent with our 2013 measurement, when METRECC did not exist. Nevertheless, given the impact of METRECC internationally, for the next step in monitoring the implemented informatica curriculum, we will try to merge METRECC's merits with our existing evaluation method. This means trying to find the delicate balance between consistency, validity and international consensus.

3   Informatics in Dutch Schools

In this chapter, we first present the current situation of the secondary school subject Informatics in the Netherlands. With this situation as a perspective, we present the results of our survey. We remark that not all the respondents answered all the questions, and conversely, some questions allowed for multiple responses. Therefore, the numbers of answers do not always correspond to the number of the respondents.

3.1   Schools and Students

The data provided by the Dutch government institution charged with the education implementation (in Dutch: Dienst uitvoering onderwijs, DUO) reveal that during the academic year 2021–2022, some 45% of both HAVO and VWO schools offered informatics with about 10% of the total HAVO student population and almost 13% of the total VWO student population choosing it [3]. According to the respondents to our survey, almost a quarter of the students are girls.

3.2   Teachers

A total of 57 informatics teachers filled in the survey, 12 of them identifying as female. Regarding their education to obtain informatics teacher qualification, 32 indicate having a university level teacher qualification, 7 followed CODI[1], course [6], and 24 mention

---

[1] In 1998, with the introduction of informatics, a consortium was set up among 12 universities and universities of applied science (in Dutch: Consortium Omscholing Docenten Informatica, CODI.) The aim was to collectively train teachers who would be responsible for implementing the informatics course in schools. During the period from 1998 to 2005, this consortium served as the exclusive pathway for individuals seeking qualification as informatics teachers.

something else, for example: still being a student, having teacher qualification for math or physics, or no teacher qualification at all. 22 of these teachers have been teaching Informatics for at least six years. Thirty of the teachers are the sole informatics teacher in their schools.

3.3 Implemented Curriculum

In this section, we report on teachers' views about the new informatics curriculum and the teaching practices they report.

Almost all the teachers are familiar with the new curriculum. Their opinions about it are shown in Table 1.

Table 1. Opinion on 2019 informatics curriculum.

What is your opinion in the 2019 informatics curriculum?	agree	somewhat agree	somewhat disagree	disagree	no opinion
It is better than the old curriculum	28	10	0	2	17
The learning goals are current and up-to-date	24	28	4	1	0
I miss topics	11	14	17	9	4
There are too many topics	16	8	18	13	2
This curriculum offers sufficient guidance for good educational practice	29	15	8	3	2

Regarding programming, 35 teachers find that it gets enough attention in the new curriculum. 37 teachers are satisfied with the topics, skills and themes contained in the new curriculum, and finally, 37 teachers are satisfied with the distribution of topics, skills and themes across the core program and the elective themes. In the comments, several teachers say they miss data science in this curriculum. Furthermore, they mention that there are too many topics, that the curriculum is too abstract and that they miss learning goals related to practical skills. Several teachers raise the question of the purpose of this Informatics subjects, and one of them says, "I would like to draw more attention to the objective of the subject and to a broader social definition of the subject (instead of the scientific definition as a starting point)."

All but three teachers feel somewhat or sufficiently equipped and supported (through available materials, refresher courses, teacher networks, etc.) to be able to shape their

teaching practice properly with the new curriculum. Some are reluctant to teach particular elective themes because they lack sufficient expertise. Many are not satisfied with the availability and quality of teaching materials and resort to writing their own. In many cases, teachers use the available teaching materials to familiarize themselves with particular elective themes.

We asked a number of specific questions regarding the elective themes: Do you feel competent enough to teach elective themes? To how many HAVO students do you teach the elective theme? To how many VWO students do you teach the elective theme? What assessment form do you use for elective themes? The teachers' answers are in Table 2. The full names of the elective themes are listed in the introduction chapter.

Table 2. Teaching practice.

|		Algorithms	Databases	Cognitive comp	Prog. Paradigms	Comp. Arch	Networks	Physical comp	Security	Usability	User experience	Impact	Comp. Science
Competent	Yes	30	49	20	31	34	25	35	32	37	34	33	31
	Somewhat	23	5	19	18	15	23	20	22	17	19	18	15
	No	4	3	18	8	8	9	2	3	3	4	6	10
HAVO	All	11	38	8	10	20	19	26	29	21	28	23	9
	Some	13	10	14	17	12	15	16	10	14	17	9	14
	None	32	8	31	27	23	22	14	17	18	12	24	32
VWO	All	27	43	17	26	24	32	34	34	25	28	24	26
	Some	11	10	13	13	16	15	14	12	15	15	10	10
	None	17	2	25	17	15	9	8	10	15	13	20	21
Assessment	Written exam	22	33	12	17	24	24	12	27	11	11	8	14
	Practical assignment	23	44	24	36	21	34	48	28	38	43	28	28
	Other	2	4	1	3	1	4	1	1	4	2	2	3
	I do not teach this	19	2	26	17	20	10	7	11	12	11	24	20

Regarding other forms of assessment, the teachers mention a *master's project*—a large (individual) practical assignment, a presentation, and a portfolio containing reports, reflection, and logbooks. When choosing the elective themes to teach, in four cases the students make a decision, in 41 cases the teacher decides, and in 12 cases the choice is made together. Individual choice for students is possible in 26 cases, while in 31 cases the choice is made for the whole class. Regarding the teaching materials for elective themes, for the HAVO the use of the textbooks is reported 29 times, 14 teachers use their own teaching materials and 4 use something else. In VWO, textbooks use is reported 65 times, teacher's own materials 41 times and something else 9 times. Table 3 indicates, per elective theme, whether the teachers teach it, and if so, whether they use the SLO

teaching materials, a commercially available textbook, or some other form of teaching materials.

Table 3. Use of teaching materials.

		Algorithms	Databases	Cognitive comp	Prog. Paradigms	Comp. Arch	Networks	Physical comp	Security	Usability	User experience	Impact	Comp. Science
Teaching materials	SLO	11	16	5	9	5	12	15	12	16	18	11	15
	Textbook	18	28	16	17	21	24	14	17	17	14	16	11
	Other	25	37	17	23	19	29	31	24	21	26	21	16
	I do not teach this	17	2	26	17	17	9	9	12	12	12	21	23

The new curriculum was developed with equity in mind and intended to be equally attractive and relevant for all students: girls and boys, students in humanities profiles as well as those in science profiles. Regarding the attractiveness of Informatics curriculum for girls, about 10% of the respondents find it not attractive. The reasons for this lay in the technical and theoretical nature of the curriculum. On the other hand, the majority of teachers, those who do not find it unattractive for girls, point out that a teacher has ample room for differentiation in their lessons and that girls tend to perform better on complex projects which require planning and taking responsibility.
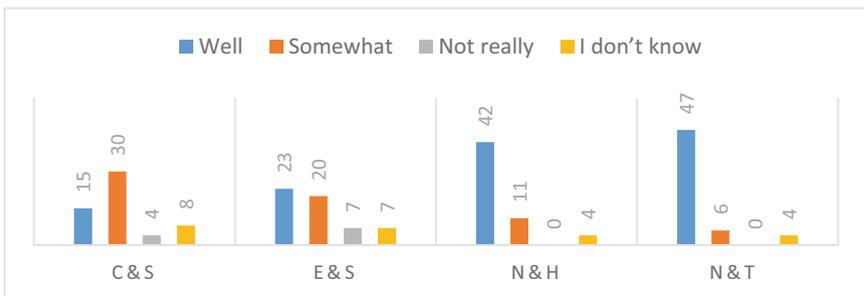
Fig. 1. Suitability of curriculum for various profiles (Culture & Society, Economy & Society, Nature & Health, Nature & Technology).

We enquired to what extent do the teachers think the elective themes do justice to the profiles. The responses are shown in Fig. 1. Some teachers comment that they miss the creative aspects of informatics such as making a website or using a 3d printer, in particular for the Culture and Society profile. Others point out that projects related to, for example, game development give students the opportunity to, "mix the coding, modelling, art, and creativity" and thus cater to the needs of students following various profiles. It is pointed out, however, that out of the twelve elective themes, eight are concerned with technical

topics and only four with the societal. Some teachers regret that they have no time to offer sufficient customization for students from all profiles. Finally, some teachers find that the curriculum does not prepare the students in the humanities profiles for their subsequent studies which require ICT skills.

With this informatics curriculum, it is possible to combine the (elective) theme in teaching practice. We asked the teachers what combinations they considered promising. The results are shown in Table 4. The row labeled total shows how many times a particular (elective) theme was mentioned. Other rows show the frequencies of particular pairings. Since some teachers mentioned combinations of three or more (elective) themes as well, some themes are counted several times when in combination with others.

Table 4. Combining (elective) themes.

	Skills	Foundations	Information	Programming	Architecture	Interaction	Algorithms	Databases	Cognitive comp	Prog. Paradigms	Comp. Architecture	Networks	Physical computing	Security	Usability	User experience	Impact	Comp. Science
Total	0	28	21	44	12	20	23	17	4	16	12	14	18	11	27	29	13	7
Skills		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Foundations			6	13	0	0	8	1	0	3	2	0	4	1	0	0	2	1
Information				8	2	1	1	9	0	1	0	0	0	0	1	0	2	1
Program					0	0	14	6	2	10	0	0	3	1	1	1	2	3
Architecture						3	0	1	0	0	2	4	1	3	0	0	0	0
Interaction							1	0	0	0	0	0	0	1	11	11	3	0
Algorithms								1	2	3	0	0	1	1	0	0	0	1
Databases									1	1	0	1	0	1	0	0	1	0
Cognitive c										1	0	0	0	0	0	0	1	0
Prog. Par											1	0	2	0	0	1	0	3
Comp. Arch												7	5	0	0	0	0	0
Networks													3	3	0	0	0	0
Phys. Comp														0	1	1	1	0
Security															0	1	2	0
Usability																20	0	2
User ex																	3	1
Impact																		0

One teacher remarked that many combinations are possible but not necessarily desirable. Another one saw no necessity to combine (elective) themes.

Finally, we enquired about the desire to introduce a national exam—a persisting and recurring issue that has been resurfacing periodically since the introduction of Informatics in 1998. Sixteen of our respondents would welcome a national exam for the core curriculum, three for elective themes, and 41 do not want it at all. In their comments, the proponents mostly argue that a national exam would help to take informatics as a school

subject more seriously by the relevant stakeholders. Furthermore, a national exam would serve as a quality control instrument—thus, serving the same purpose it has for all the school subject which do have a national exam. The opponents mention that a written exam is contrary to the nature of informatics where student mostly do practical work. They expect that teachers would start *teaching to the test*—a practice widely observed in subjects with national exam. Most of all, they point out that without a national exam, there is a lot of freedom to choose what to teach and how to do it (often in the form of large group projects) which provides ample room for differentiation and to cater to students' needs. Finally, the opponents address the issue of teacher quality and their scarcity, expressing concern that a significant number of teachers might stop teaching informatics. Consequently, schools would cease to offer the subject, potentially leading to a negative impact on the national level.

4   Discussion and Conclusion

This paper explores the situation of the secondary school subject Informatics in the Netherlands, and in particular the implementation of the new Informatics curriculum from the Informatics teachers' point of view.

4.1   Schools and Students

We see that there is a decline in the number of schools offering Informatics, while the total student numbers seem to stabilize. Furthermore, there is an increasing number of schools employing more than one Informatics teacher (c.f. [5, 6]). A number of schools recognize the importance of Informatics education, offer programming classes in lower secondary grades and actively promote Informatics in the upper secondary grades. Consequently, such schools employ more than one Informatics teacher. Some other schools, with small numbers of students choosing Informatics, chose to discontinue the Informatics course for financial reasons. Finally, the general teacher shortage in the Netherlands, and in particular shortage of (qualified) Informatics teachers, regularly compels schools to either employ an underqualified Informatics teacher, or to discontinue offering Informatics altogether.

4.2   Teachers

A significant proportion of our respondents—those with no Informatics teacher qualification and those with CODI qualification—have a field of initial study other than computing. Similar results are reported about Informatics teachers in French-speaking Switzerland [8] signifying that the Netherlands is not unique in this aspect.

We asked the teachers their opinion about the new 2019 Informatics curriculum. Seventeen out of 57 teachers have no opinion on whether it is better that the old one. None of these teachers have been teaching for longer than six years and it is plausible that they never taught Informatics according to the 2007 curriculum. The majority of the teachers find that the curriculum is modern and up to date, which was stated as one of the goals when it was being revised. However, the topic of data science is absent

while it could be argued that it is becoming an essential part of any modern Informatics curriculum. A number of teachers mention the curriculum is too broad, too abstract, or lacking learning goals related practical skills. Yet, the majority of teachers state that it offers sufficient guidance for good educational practice. When interpreting these results, we realize that some teachers see the Informatics curriculum as intended curriculum [11] (formal curriculum, i.e., learning standards prescribed by law), while others equate the curriculum to its interpretation in textbooks. The entire curriculum contains 18 themes, and the first one, Skills, is intended to be taught only integrated with others. Consequently, a HAVO student would need to learn about eight (elective) themes (six in core curriculum plus two electives) and a VWO student about ten (elective) themes. Furthermore, all but one teacher see chances to combine teaching of (elective) themes.

It is therefore somewhat surprising to realize that some teachers perceive the curriculum as overloaded, despite its original intention of addressing overload through elective themes. A further remark concerns the abstract nature of the curriculum. It was the intention of this curriculum to offer the essence of computing as scientific discipline and express the learning goals on a rather abstract level, at the same time trying to be *future proof* (as much as possible in a discipline characterized by rapid developments) [1, 10]. In line with the Dutch tradition, it is up to teachers—and in practice, often up to textbook writers—to interpret the curriculum. The remark about missing practical skills can be seen in the same light.

The teachers reported on how well they felt equipped and supported to shape their teaching practice. The issues they mention are present since the introduction of Informatics in 1998: those not fully qualified might not possess the necessary expertise, and the teaching materials vary greatly in quality or are even non-existent for certain elective themes [5, 6, 9]. Fortunately, the development of SLO teaching materials for elective themes was in almost all the cases accompanied by free courses on corresponding topics which provided the opportunity for teachers to familiarize themselves with, for example, Physical Computing, or agent-based modeling for the elective theme Computing Science.

Regarding the elective themes taught, we see that Databases score the highest. This finding confirms that the decision to include this topic into the new Informatics curriculum was right, and prompts for the discussion to include it into the core curriculum, should the Informatics curriculum be revised in the future. Physical Computing scores high as well, and we assume that this fact is due the high availability of Arduino, micro-bit and similar platforms, and extensive availability of teaching materials and online courses. Some elective themes are not taught often. Algorithms, computability and logic; and Computer architecture might be perceived as technical and abstract. The same holds possibly for Cognitive computing, which is about artificial intelligence and machine learning. What these three elective themes have in common is that there are scarce teaching materials and that SLO materials have just recently became available. Social and individual impact of informatics scores low as well. This theme is not technical at all and may therefore feel unfamiliar to a computer scientist or Informatics teacher. Furthermore, it reflects Dutch societal norms and values and therefore—contrary to many technical themes—depends on Dutch teaching materials. Finally, Computational Science is about modeling and simulation. Again, a computer scientist or Informatics

teacher is not necessarily familiar with this topic. The SLO teaching materials implement modeling and simulation through the use of NetLogo software and agent-based modeling paradigm, which is possibly new to many Informatics teachers.

All teachers but one see opportunities to combine (elective) themes, which is in line with the intentions of the new curriculum. Many (elective) themes share certain aspects: for example, developing an agent-based model in NetLogo for the elective theme Computational Science means using a programming paradigm other than imperative (mentioned in theme Programming in core curriculum), and thus implies elective theme Programming Paradigms. Programming (in the core curriculum) is most frequently mentioned theme suitable to combine with other themes, and that is not surprising, considering that programming plays a role in the production of just about every digital artifact. Usability and User experience are next, and the combination of these two is the most frequently mentioned combination. The themes taught less frequently are also less frequently mentioned as candidates for combined teaching. Extensive analysis of this aspect goes beyond the scope of this paper, but it would be interesting to further explore teachers' views on this: for example, how come no one mentions the combination of Programming and Physical Computing?

Regarding the introduction of a national exam, 72% of our respondents are against it. We obtained similar results in our 2013 survey, and even in a survey preformed in 2007, about 55% of the respondents were against a national exam, and some 28% were in favor [9]. Throughout all these years, proponents and opponents keep mentioning similar reasons in favor or against a national exam.

4.3 Answers to the Research Questions

With the analysis of the data above, we are now able to answer our research questions.

1. What are the experiences of Dutch informatics teachers with the new informatics curriculum? In general, the teachers share positive experiences. As we saw in 2013, teachers do not consider themselves to be completely competent for teaching the curriculum. For instance, regarding the elective themes, there is quite some variation. Regarding the theme Cognitive computing, 34% of the teachers feel they lack competence, while this is the case for just 5% when it comes to the theme Databases. We see that the teachers obviously feel more competent regarding the traditional themes than with respect to the new ones. Approximately 75% of the assessments are in the form of a practical assignment, and 25% in the form of a written exam, which in our view is justified by the dominantly practical character of the subject.
2. To what extent are Dutch informatics teachers satisfied with the new informatics curriculum? The vast majority (about 88%) of the teachers is positive about the new curriculum. Nevertheless, they consider this curriculum clearly to be more suitable for students in the nature profiles than for those in the society profiles. Approximately the same proportions of the teacher population consider the range of elective themes to be either too narrow or too broad.

On the most global level, we conclude that the new informatics curriculum is a substantial step in the good direction, but that we need to work on fixing the bias towards the suitability for the nature profiles in order to really make Informatics 'an all-around

subject". Furthermore, the professional development of teachers, as was pointed out in 2013, is still a huge issue.

References

1. Barendsen, E., Grgurina, N., Tolboom, J.: A new informatics curriculum for secondary education in The Netherlands. In: Brodnik, A., Tort, F. (eds.) ISSEP 2016. LNCS, vol. 9973, pp. 105–117. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46747-4_9
2. Benaya, T., et al.: Computer science high school curriculum in Israel and Lithuania – comparison and teachers' views. Balt. J. Mod. Comput. **5**, 164–182 (2017)
3. DUO: Examens Bestanden over examenkandidaten onderverdeeld in diverse variabelen., https://duo.nl/open_onderwijsdata/voortgezet-onderwijs/aantal-leerlingen/examens.jsp, (2023)
4. Falkner, K., et al.: An international study piloting the measuring teacher enacted computing curriculum (METRECC) instrument. In: Annual Conference on Innovation and Technology in Computer Science Education, pp. 111–142. ITiCSE (2019)
5. Grgurina, N., et al.: The second decade of informatics in Dutch secondary education. In: Pozdniakov, S.N., Dagienė, V. (eds.) Informatics in Schools. Fundamentals of Computer Science and Software Engineering, pp. 271–282. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-030-02750-6_21
6. Grgurina, N., Tolboom, J.: The first decade of informatics in Dutch high schools. Inform. Educ. **7**(1), 55–74 (2008)
7. KNAW: Digitale geleterdheid in het voortgezet onderwijs. Koninklijke Nederlandse Akademie van Wetenschappen (2012)
8. Parriaux, G., Pellet, J.-P.: Computer Science in the Eyes of its Teachers in French-Speaking Switzerland. In: International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, pp. 179–190 Springer (2016). https://doi.org/10.1007/978-3-319-46747-4_15
9. Schmidt, V.: Vakdossier 2007 informatica (Dossier of the Subject Informatics), Tech. Rep. Enschede Netherlands. SLO (2007)
10. Tolboom, J., et al.: Informatica in de bovenbouw havo/vwo: Naar aantrekkelijk en actueel onderwijs in informatica. SLO (2014)
11. Van den Akker, J.: Curriculum perspectives: an introduction. In: Curriculum Landscapes and Trends, pp. 1–10 Springer Netherlands, Dordrecht (2003). https://doi.org/10.1007/978-94-017-1205-7_1

Navigating the Implementation
of the Curriculum Digital Education
in Austrian Secondary Schools:
Challenges and Teacher Perspectives

Corinna Hörmann[ID], Eva Schmidthaler(✉)[ID], and Barbara Sabitzer[ID]

Johannes Kepler University Linz, Altenbergerstraße 69, 4040 Linz, Austria
{corinna.hoermann,eva.schmidthaler,barbara.sabitzer}@jku.at

Abstract. For a considerable period, there was only a single year of compulsory Computer Science (CS) education in Austria that was required as part of a student's academic career. The mandatory curriculum "Digital Education" (Digitale Grundbildung) was launched in September 2018 for all students in lower secondary education, formally integrating 21st-century skills into upper grades. The school's management could choose to offer "Digital Education" as a stand-alone course or to include it in other classes. The 2022/2023 academic year saw the addition of an altered curriculum, where the subject was added to the normal timetable. Nonetheless, schools continue to deal with the problem of who is teaching what and how due to a staffing shortfall and a lack of instructional materials. This paper reports on experiences with the 2022 curriculum by evaluating a survey that examines the implementation of the mandatory curriculum "Digital Education" in Austrian secondary schools and the challenges that teachers face in navigating this new landscape. Aiming to do so, the qualitative data were interpreted, summarized, and discussed, whereas the quantitative data were analyzed in former papers. Despite the obstacles that emerged, teachers recognized the importance of digital literacy skills for their students and expressed a desire for more support in implementing the curriculum. The findings of this study have implications for policymakers, school administrators, and teacher training programs as they work to ensure the successful integration of "Digital Education" in secondary schools.

Keywords: Digital Education · 21st Century Skills · Curriculum · STEM

1 Introduction

Around 90% of today's employment require the most basic IT skills, according to relevant studies conducted by the EU Commission for the EU Digital Agenda [14]. A child who understands how to code will also have more work opportunities in the technology, finance, retail, and health industry, as well as

various other areas [24]. Therefore, the European Digital Competence Framework (DigCompEdu) responds to the need that every European citizen should gain necessary competencies to use digital technologies critically and creatively. It provides a structure to understand what it means to be digitally competent and gives a sound background that can guide policies in different countries [23].

Researching a new curriculum, like "Digital Education" in Austria, allows for an assessment of its effectiveness in enhancing educational outcomes. By identifying strengths and weaknesses, adjustments can be made to improve the curriculum and the educational experience for students and teachers. Furthermore, data and findings can guide educators and policymakers in making informed decisions about curriculum design, implementation, and modifications. However, researching a new curriculum promotes a culture of continuous improvement in education. It also motivates educators to evaluate their methods, spot opportunities for development, and implement evidence-based adjustments to boost both curriculum and instruction. Continuous development is essential for ensuring that education remains dynamic, responsive, and effective in preparing students for the changing demands of the modern world.

This paper reflects on experiences from teachers by analyzing the qualitative data from a study that focuses on the implementation of the mandatory curriculum "Digital Education" in Austrian secondary schools. Chapter two describes the theoretical background by concentrating on the history of the subject in Austria. The study with its methodology, results, and discussion is described in section three, whereas chapter four provides a conclusion and an outlook on upcoming work.

2   Theoretical Background

In 1985, Austria made Computer Science (CS) a mandatory standalone subject in schools, with two hours a week in the 9th grade [13,16]. In 2018, the mandatory subject "Digital Education" was introduced in lower secondary education (grades five to eight) in Austria [2], covering digital competences, media competences, and civic education [15]. Furthermore, the Austrian government published a masterplan for digitalization, including three sub-projects: revising existing curricula, teacher training (TT), and expanding technical infrastructure [1,19].

Moreover, an 8-point-concept was presented to foster digital education, which included a platform called Portal Digital School (PoDS) [9], Massive Open Online Courses (MOOCs) [8], and the Eduthek [5]. Additionally, the installation of standardized qualification marks in the evaluation and certification of learning applications was planned and has been launched already [7]. Furthermore, basic IT infrastructure is being improved in federal schools [6]. Funding for digital devices for students was planned for the 5th and 6th grades in 2021/22, but shipping problems caused delays [3,4,18]. However, since the school year 2023/24, every child has access to their own learning and teaching device.

In March 2022, the concepts of a new curriculum were presented by the Ministry of Education by implementing the 4C's of the 21st century: *Critical*

Thinking, *Creativity*, *Collaboration*, and *Communication* [10]. A two-dimensional competence model forms the basis of the presented curriculum of "Digital Education". The different areas of competencies form the horizontal line: (1) *orientation* – analyzing and reflecting on social aspects of media change and digitization, (2) *information* – responsible handling of data, information, and information systems, (3) *communication* – communicating and cooperating using media systems, (4) *production* – creating and publishing digital content, designing algorithms, and creating software programs, (5) interaction – responsible use of offers and options of a digital world [10]. The vertical classification describes the subject-specific topics that are represented in the "Frankfurt Dreieck" [11]. This theory can be seen as an extension of the famous "Dagstuhl-Dreieck" but concentrates on the aspects of digital education. The three central concepts are based on the following perspectives: (T) *technical-media* – structures and features of digital, IT, and media systems, (G) *social-cultural* – social interactions through the use of digital technologies, and (I) *interaction-related* – interaction in the form of usage, action, and subjectification [11].

One annual hour each week is implemented in the curriculum's redesigned model from grades 5 to 7 starting in the school year 2022/23, with year 8 following in the consecutive school year [18, 22]. The new competence-oriented curriculum will be installed at the beginning of the next school year 2023/24 in both primary level and secondary level and therefore, "Digital Education" will be compulsory for all Austrian primary and secondary school students [18].

Another issue arose with the introduction of the new subject because there are presently no study programs for "Digital Education" in Austrian teacher preparation programs like there are for other traditional subjects. Postgraduate teacher training began in the fall of 2022 to address the field's need of fully qualified staff. Still, there is a lack of vacant spots at these programs, as a lot of teachers want to join. Nevertheless, Gabriel et al. state that adequate teacher training is crucial to ensure digital policies are adopted and therefore implemented in the classroom. The PISA winners Estonia, Finland, Germany and New Zealand all set a strong focus on teacher training with improvement of teachers' digital skills [17].

3   Study

3.1   Methodology

The underlying study concentrated on the implementation of Austria's mandatory "Digital Education" curriculum, which went into implementation in September 2022. Teachers who actually teach "Digital Education" in the current school year 2022/23 received the survey, which was delivered to all secondary public schools in Austria. In total, 795 teachers agreed to start the survey – only 673 were able to finish it. The complete questionnaire is listed in the appendix.

To distinguish the participants from other teachers, it was first confirmed that they truly teach "Digital Education" throughout the current school year.

The second part of the survey concerned gender, age group, years in service, school type, and subjects taught.

The next section focused on the teacher training course "Digital Education", whereas the final segment focused on the teachers' individual experiences.

Additionally, the question "I would also like to say the following" gave participants the chance to add their own opinions. Ninety-eight people completed this section. The focus of this work lies on analyzing the qualitative survey data, whereas previous articles have looked at the quantitative survey data [19].

By performing a qualitative content analysis, the gathered qualitative data was evaluated with regard to the seven steps of the spiral model (see Fig. 1) of Kuckartz and Rädiker [20]. Overall, the image provides an overview of the key steps involved in conducting a structured qualitative content analysis: In the first of the seven phases, the text has to be analyzed, structured and summarized. In the next step the main categories need to be identified and the first coding process takes place, regarding those categories. If need be, sub-categories can be developed and the second coding process is conducted. After that various analysis may be performed, whereas the last step is defined by documenting the process and the results. Of course, it is possible, and often necessary, to restart the spiral-process again [20].
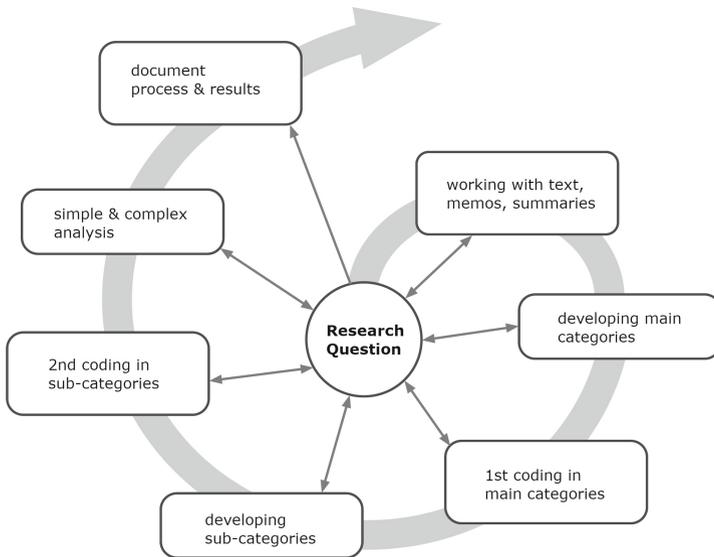
Fig. 1. Sequence of a qualitative content analysis in seven phases [20] (edited by the authors)

Transcribing the text was the first stage of the evaluation process. Existing dialects have been altered and rewritten to improve compatibility. Step two involved paraphrasing the words and identifying important information. The

statements were subsequently arranged in accordance with several categories, although one statement might possibly be related to multiple topics. We defined five key codes to identify the major problems teachers described. Problems dealt with "large groups of students", "lack of skills of teachers", "resources", "teacher training", and "curriculum". Furthermore, the statements were categorized in three codes, whether or not teachers support the new curriculum: "positive", "neutral", and "negative".

The computer-supported evaluation was done using the software "MAXQDA" and a code book, though more codes might be added if necessary. The fourth and final part of the review involved finding potential arguments by fusing qualitative and quantitative analyses.

The online tool "LimeSurvey" was utilized to gather data. This application was provided by the university for free and is GDPR (General Data Protection Regulation) compliant. Moreover, it ensures total data export, as well as in-built data evaluation [21].

Ethics were taken into account during the entire study. Introducing a new curriculum may appear innocent to a researcher at first glance but can be highly sensitive to the participating teacher or other involved parties. As the introduction of "Digital Education" was strongly politically influenced, it was vital that respondents remain anonymous to reduce undesirable consequences at all costs. The inability to track participants was also crucial, particularly when identifying people through data backtracking was minimized as effectively as possible [12].

3.2   Quantitative Results

The quantitative results have been published before but are worth summarizing (see Fig. 2 and 3), as the combination of both, quantitative and qualitative, parts are discussed later. To conclude, 90.1% of the respondents stated that they "strongly agreed" or "agreed" that they prefer the curriculum as a stand-alone subject. This hypothesis was supported by the control question, "I think it was better when "Digital Education" could still be integrated into other subjects", with 67.9% "disagreeing" or "strongly disagreeing". There is only one area in the curriculum that sticks out when compared to others. Fifty-five percent of the participants rated their knowledge in the field of "creating and publishing content digitally, designing algorithms, and programming" with "intermediate"to "very poor", which is alarming, considering the fact that those teachers already implement the curriculum [19].

3.3   Qualitative Results

Ninety-eight out of 673 participants (14.6%) completed the voluntary comment section. The text field had an average character count of approximately 316, with the shortest comment having only four words and the longest text consisting of 216 words.
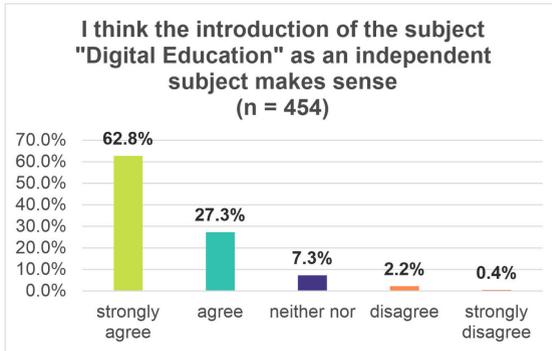
Fig. 2. Quantitative result "I think the introduction of the subject "Digital Education" as an independent subject makes sense" [19]
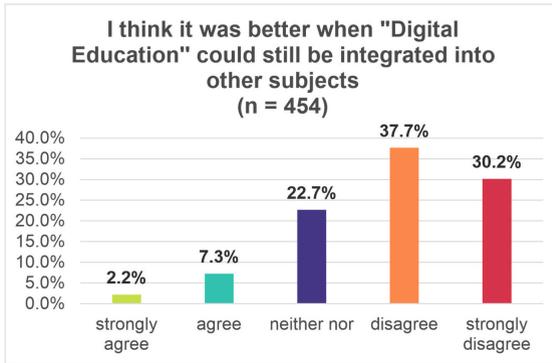

Fig. 3. Quantitative result "I think it was better when "Digital Education" could still be integrated into other subjects" [19]

Figure 4 lists the open comments' most frequent words, whereas "digital" was stated seven, and "education" four times.

When looking at the attitude of teachers towards the curriculum of "Digital Education", 30 statements could be identified as "positive", 39 as "negative", and 42 as "neutral", which is displayed in Fig. 5. Of course, it was also possible that one comment was assigned more than one code, as the descriptions of the teachers often consisted of more than one argument.

Performing a category-based quantitative analysis one can look at the key codes related to problems to determine the type of difficulty teachers face with the subject (see Fig. 6).

In total, 67 problems were listed. Five (7.5%) teachers stated that they had major problems because of a "large group" of students. They described that "they urgently need class divisions" and that "it is not possible to teach "Digital Education" with over 25 students seriously". Nine (13.4%) participants
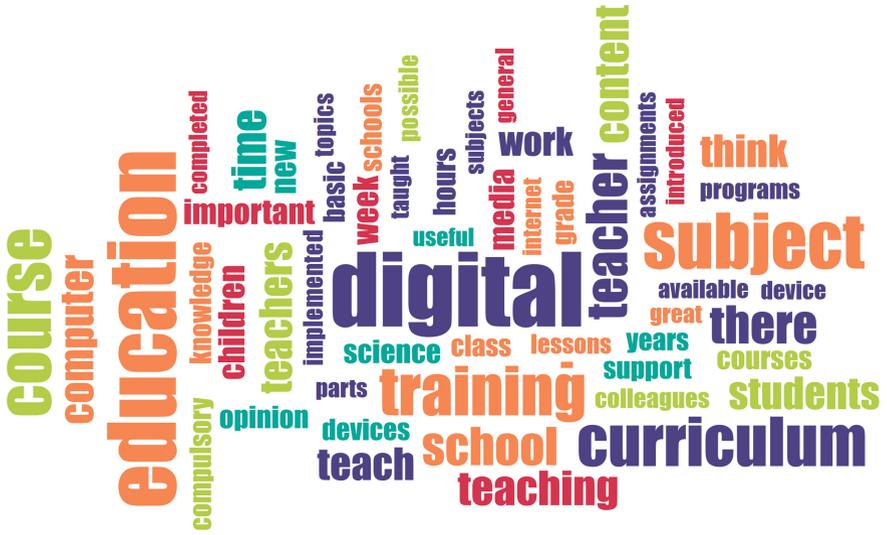
Fig. 4. Wordcloud of most common words
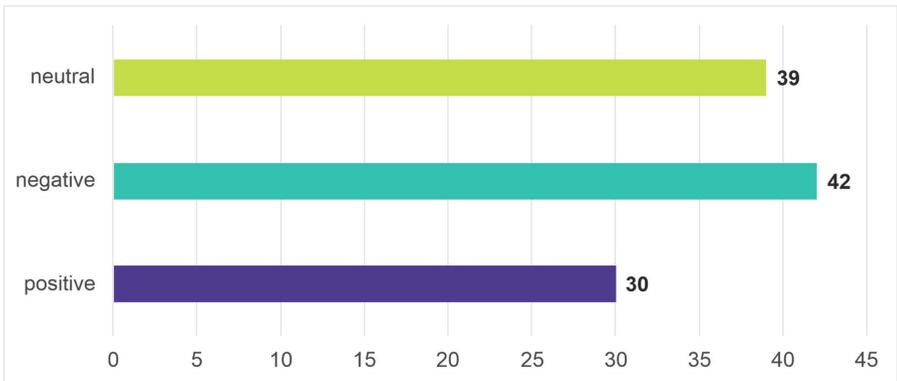
Fig. 5. Attitude towards the subject "Digital Education" (n = 98)

stated there are few or non-suitable "resources" for teaching "Digital Education". Among the comments were statements like "recently received the textbook and I find it linguistically far too complicated", "a good suitable book(s) would be good", "existing textbooks are too theory-heavy and complicated", "the content for teaching has to be gathered from countless pages on the Internet", or "an official platform with sample lessons and networking opportunities would help a lot". Another eleven (16.4%) saw the difficulty within the lack of digital skills of teachers, stating "I do not know anything about it – how should I, without training", "teachers do not have the necessary background knowledge for this subject", "there was no training for it yet", "computer scientists should teach",
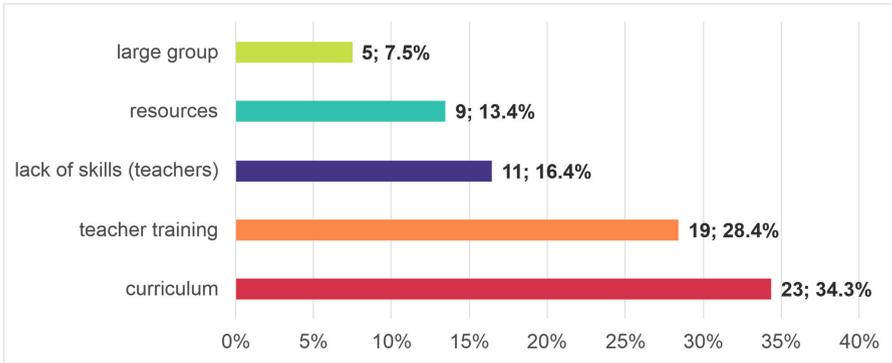
Fig. 6. Stated problems of teachers concerning the subject (n = 98)

"I always have to acquire certain knowledge by myself before I can teach it", "I do not understand what is expected of me, what I am supposed to do", and "little or no technical understanding". Nineteen (28.4%) saw a problem in the official "teacher training", saying "you can't take a teacher training course for everything", "I lack the basics of IT to support the students in understanding this. This was not an issue in the course.", "It would need more places in the teacher training course", "I didn't get a fixed place", "training for teaching staff should have started much earlier", "there is far too much theory involved", "they should provide more study places. 50 seats for 700 schools is an imposition!", "some parts are unfortunately not ideally planned", "teacher training course is not doing anything for me at the moment", or "teacher training course urgently needs to be designed more professionally". When looking at the curriculum itself, 23 (34.3%) described obstacles, stating "a lot of material in little time", "not implementable", "curriculum is too general and not helpful", "level of the students is not compatible with the curriculum", "curriculum is far from any reality", "curriculum as it is needs urgent revision", "it consists of word shells and management blah", and "clearly created by theorists".

We discovered that some codes are frequently used together while others stand alone entirely when looking at a code-relation-model (see Fig. 7), which depicts how close the key codes are to each other in different statements.

Problems with "resources" seam to have a connection to "curriculum', showing a frequency of three, as well as to "teacher training" with two connections, and "lack of skills of teachers" with one connection. The issue of "lack of skills of teachers" connect to "curriculum" problems with a frequency of three. Furthermore, "teacher training" and "curriculum" have one connected argument. Interestingly, problems with "large groups" of students do not interact with other occurring issues.

Negative classified comments show a high number of 18 connections with the problem "curriculum", eight with "teacher training", five with "lack of skills of teachers", four concerning "resources", and four "large groups". On the con-
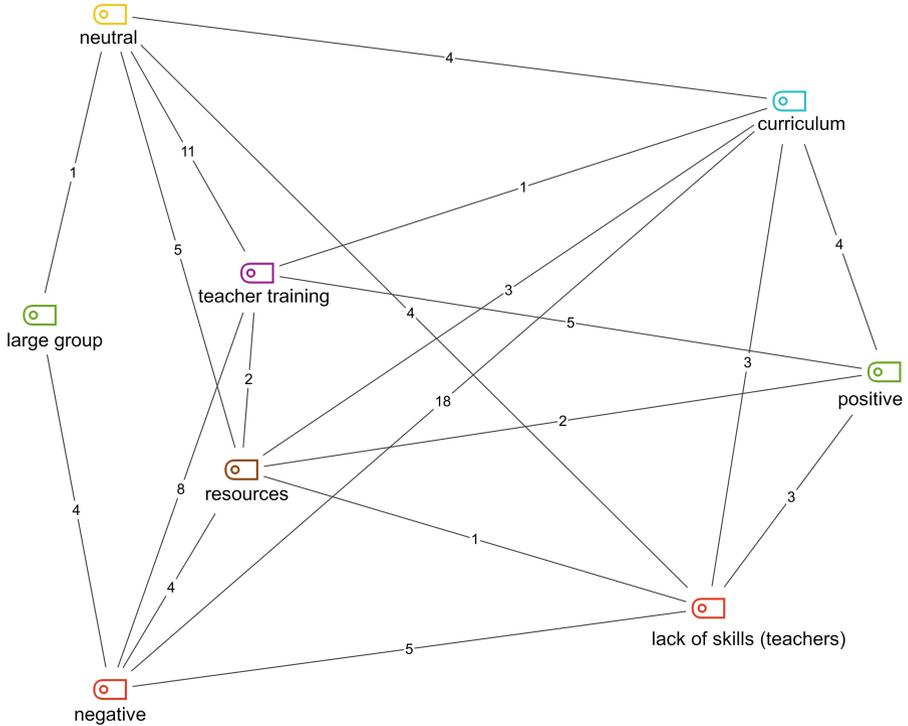
Fig. 7. Code-Relations-Model (n = 98)

trary, positive arguments connect to the problems of "teacher training" with a frequency of five, "curriculum" with four, "lack of skills of teachers" with three, and "resources" with two relations. When taking a look at neutral comments, eleven could be connected with "teacher training", five with "resources", four with "lack of skills", another four with "curriculum", and one with "large group of students".

3.4   Discussion

The underlying survey depended on volunteer samples because visiting educators at school or conducting experiments in class was impossible due to the COVID pandemic. Of course, this sampling carries the possibility that some potential participants may opt-out. Nevertheless, this strategy was met since it was the only method of contacting Austrian teachers. When looking at the rating of the subject "Digital Education" in the comment section, negative and neutral clearly dominate. This could also be due to the fact that most people fill out the comment section when they have something negative to add. Furthermore, a study (n = 117) from the authors from 2020 found out that 61% of the participating teachers rated the subject "very useful", 26% "rather useful", 9% "partly

useful", 2% "rather not useful", and 2% "not useful at all". The rating was in the upper section with an arithmetic mean of approximately 1.57.

A total of 39% stated problems with the curriculum itself, even if the analysis of the quantitative data showed that the question "I think the content of the curriculum for the subject "Digital Education" makes sense" was answered by 42 (9.3%) with "strongly agree", by 211 (46.5%) with "agree", by 122 (26.9%) with "neither agree nor disagree", by 67 (14.8%) with "disagree", and by twelve (2.6%) with "strongly disagree". The median lies with "agree", whereas the arithmetic mean can be found at 2.55 (when numbering the Likert scale from one to five downwards) [19].

Furthermore, similarities can be found in qualitative and quantitative data when the "lack of skills of teachers" were described. In the comment section, 18.6% of the participants observe problems with some parts of the curriculum, mostly technical issues.

Interestingly, teachers also mentioned that they have to teach large groups of students, as currently students are not divided into two groups with two instructors in "Digital Education" class, as it is the case in Computer Science.

4   Conclusion and Outlook

A new curriculum's introduction is a continual process that calls for ongoing improvement. Knowing how teachers felt about the initial implementation will identify areas that require enhancement and guide modifications to subsequent roll-outs. The underlying research can be used to determine areas where teachers might want additional training or support by understanding how they responded to the new curriculum. This could result in better opportunities for teachers to advance their careers, which would ultimately improve curriculum delivery.

To summarize, most teachers stated problems with "large groups", "lack of skills", "resources", "teacher straining", and "curriculum", when talking about "Digital Education" in Austria. Still, when combining both quantitative and qualitative data it stands out that most teachers approve of the new subject and its stand-alone version over the old one, but a major knowledge gap of teachers can be observed in the field of "creating and publishing content digitally, designing algorithms, and programming".

With the introduction of the compulsory subject another problem appeared, as currently no entire "Digital Education" study courses in Austrian teacher education exist, as there is for other traditional subjects. In autumn of 2022, postgraduate training for teachers started to tackle the lack of fully trained staff in "Digital Education". Henceforth, there is still much effort that has to be put into implementing the curriculum to gain further teachers' motivation. To help educators dealing with the unfamiliar curriculum of "Digital Education", it is necessary to create an extensive collection of material for the specific topics, especially for the technically oriented ones, for example, in the field of "creating and publishing content digitally, designing algorithms, and programming".

Moreover, a new study started to find out the best possible options for teacher training.

A   Appendix

A.1   Questionnaire

1. I am teaching the subject "Digital Education" in the current school year 2022/23: yes/no
2. Gender: male/femal/diverse
3. Age: under 30/30–39/40–49/50–59/60+
4. Years in service: under 5 years/5–20 years/11–20 years/21–30 years/30+ years
5. I teach in the following school type(s): secondary school/lower level AHS/higher level AHS/other
6. subjects: Vocational Orientation/Physical Activity and Sports/Fine Arts Education/Biology/Chemistry/(Descriptive) Geometry or Geometric Drawing/German/Digital Education/English/Nutrition and Household/French/ Geography/History/Computer Science/Italian/Latin/Mathematics/Music Education/Physics/Political Education/Psychology & Philosophy/Religion/ Spanish/Technical Work/Textile Work/other
7. Are you currently attending the teacher training course for "Digital Education"? yes/no
8. If no at (7): Are you planning to attend such a course in the future? yes/maybe/no
9. If no at (8): Why is such a course out of the question for you? no time/I already know everything about it/no interest/too much work/not supported by the school/other
10. Please rate the following statements (strongly agree, agree, neither agree nor disagree, disagree, strongly disagree):
    – I think the content of the curriculum for the subject "Digital Education" makes sense.
    – I think the introduction of the subject "Digital Education" as an independent subject makes sense.
    – I think it was better when "Digital Education" could still be integrated into other subjects.
    – I am having troubles preparing for "Digital Education" class.
    – I have sufficient resources to prepare for lessons in "Digital Education".
    – I feel confident in terms of content in "Digital Education" class.
    – I think that "Digital Education" should be taught by teachers who studied computer science.
11. Please rate your knowledge in the individual competence areas of the "Digital Education" curriculum based on school grades:
    – analyzing and reflecting on social aspects of media change and digitization
    – handle data, information, and information systems responsibly

- communicating and cooperating using IT systems
- creating and publishing content digitally, designing algorithms, and programming
- assess offers and options for a world shaped by digitization and use them responsibly

12. Would you like to have more support in implementing the "Digital Education" curriculum?
13. If yes at (12): Which offers would you use? teacher training at universities/teacher training at school/online teacher training/online resources/books/
other
14. I would also like to say the following: ...

References

1. BMBWF: Masterplan für die Digitalisierung im Bildungswesen (2018). https://www.bmbwf.gv.at/Themen/schule/zrp/dibi/mp.html
2. BMBWF: Verbindliche Übung Digitale Grundbildung - Umsetzung am Schulstandort (2018). https://www.bmbwf.gv.at/Themen/schule/zrp/dibi/dgb.html
3. BMBWF: Digital devices for students - Digitale Schule - Bundesministerium für Bildung, Wissenschaft und Forschung (2020). https://digitaleschule.gv.at/digitale-endgerate-fur-schulerinnen-und-schuler/
4. BMBWF: Digital devices for teachers - Digitale Schule - Bundesministerium für Bildung, Wissenschaft und Forschung (2020). https://digitaleschule.gv.at/digitale-endgerate-fur-lehrerinnen-und-lehrer/
5. BMBWF: Eduthek - Digitale Schule - Bundesministerium für Bildung, Wissenschaft und Forschung (2020). https://digitaleschule.gv.at/ausrichtung-der-eduthek-nach-lehrplanen/
6. BMBWF: Expansion of the school's basic IT infrastructure - Digitale Schule - Bundesministerium für Bildung, Wissenschaft und Forschung (2020). https://digitaleschule.gv.at/ausbau-der-schulischen-basis-it-infrastruktur/
7. BMBWF: Learning Apps - Digitale Schule - Bundesministerium für Bildung, Wissenschaft und Forschung (2020). https://digitaleschule.gv.at/gutesiegel-lernapps/
8. BMBWF: MOOC - Digitale Schule - Bundesministerium für Bildung, Wissenschaft und Forschung (2020). https://digitaleschule.gv.at/lehrenden-fortbildung/
9. BMBWF: PODS - Digitale Schule - Bundesministerium für Bildung, Wissenschaft und Forschung (2020). https://digitaleschule.gv.at/portal-digitale-schule/
10. BMBWF: Projekt Lehrpläne NEU - Fachlehrplan Digitale Grundbildung (2022). https://www.ahs-informatik.com/app/download/10356599585/22-03-14_Fachlehrplan_Digitale+Grundbildung_sek+I.pdf?t=1651008169
11. Brinda, et al.: Frankfurt-Dreieck zur Bildung in der digital vernetzten Welt - Ein interdisziplinäres Modell (2019)
12. Cohen, L., Manion, L., Morrison, K.: Research Methods in Education. Routledge, London (2018)
13. Döbeli, B.: ICT im Hosensack - Informatik im Kopf? In: Brandhofer, G., Futschek, G., Micheuz, P., Reiter, A., Schoder, K. (eds.) 25 Jahre Schulinformatik in Österreich. Zukunft mit Herkunft, pp. 35–44 (2010)
14. European Commission: Digital Agenda for Europe. https://eige.europa.eu/resources/digital_agenda_en.pdf

15. Ferrari, A.: Digital competence in practice: an analysis of frameworks. In: Joint Research Centre of the European Commission, p. 91 (2013). https://doi.org/10.2791/82116
16. Friedrich, S., Hartmann, W.: Informatikunterricht im Spannungsfeld zwischen Tastendruck und UML. In: Brandhofer, G., Futschek, G., Micheuz, P., Reiter, A., Schoder, K. (eds.) 25 Jahre Schulinformatik in Österreich. Zukunft mit Herkunft, pp. 27–28 (01 2010)
17. Gabriel, F., Marrone, R., Sebille, Y.V., Kovanovic, V., de Laat, M.: Digital education strategies around the world: practices and policies. Irish Educ. Stud. **41**, 85–106 (2022). https://doi.org/10.1080/03323315.2021.2022513
18. Hörmann, C., Schmidthaler, E., Kuka, L., Rottenhofer, M., Sabitzer, B.: From non-existent to mandatory in five years - the journey of digital education in Austrian schools. In: International Conference on Informatics in Schools (ISSEP) (2022)
19. Hörmann, C., Schmidthaler, E., Sabitzer, B.: Introducing digital education as a mandatory subject - the struggle of the implementation of a new curriculum in Austria. In: Proceedings of the 15th International Conference on Computer Supported Education - Volume 2: CSEDU, pp. 213–220. INSTICC, SciTePress (2023). https://doi.org/10.5220/0011837000003470
20. Kuckartz, U., Rädiker, S.: Qualitative Inhaltsanalyse. Methoden, Praxis, Computerunterstützung. Beltz Juventa, 5 edn. (2022)
21. Limesurvey: Free university surveys & questionnaires. https://www.limesurvey.org/solutions/universities
22. Polaschek, M.: Erledigung BMBWF - Österreichisches Parlament - 9338 (2022). https://www.parlament.gv.at
23. Redecker, C., Punie, Y.: European framework for the digital competence of educators: DigCompEdu. Publications Office of the European Union (2017). https://doi.org/10.2760/159770, https://moodle.ktu.edu/pluginfile.php/428841/mod_resource/content/1/pdf_digcomedu_a4_final.pdf
24. Videnovik, M., Vlahu-Gjorgievska, E., Trajkovik, V.: To code or not to code: introducing coding in primary schools. Comput. Appl. Eng. Educ. **29**, 1132–1145 (2020)

Bridging the Gap: Infusing Natural Science Classes with Computer Science Concepts and Skills

Elena Yanakieva[1]([✉]) [iD], Annette Bieniusa[1] [iD], Thomas Becka[3],
Brian B. Moser[1,2] [iD], Dominik Jerger[1], and Christoph Thyssen[1]

[1] University of Kaiserslautern-Landau, Kaiserslautern, Germany
{elena.yanakieva,annette.bieniusa,christoph.thyssen}@rptu.de
[2] German Research Center for Artificial Intelligence (DFKI),
Kaiserslautern, Germany
brian.moser@dfki.de
[3] Eduard-Spranger-Gymnasium, Landau, Germany

Abstract. As we venture further into an ever-evolving digital society, it has become imperative for schools to adapt and equip future generations with the required skills. Computational Thinking (CT), specifically its algorithmic thinking aspect, plays a significant role in computer science and other sciences, such as biology, chemistry, and physics. It encourages students to reason at multiple levels of abstraction. This ability is particularly useful in natural science, as scientific experiments require critical skills like conceptualization, problem decomposition, and solving or designing structured systems. However, due to organizational and curricular restrictions, these concepts and skills are typically taught separately, leaving it to the learners to connect and apply the acquired skills in the respective context.

To bridge this gap, we designed a two-day workshop that embeds CT in a physics and sustainability context, namely energy-efficient housing. In this workshop, we employ the Calliope Mini, a micro-controller explicitly designed for educational purposes, to teach essential algorithmic thinking and data processing. We carried out the workshop twice in a primary school in Germany, each with 20 children in 4th grade (ages 9–10). In this experience report, we present the multi-disciplinary workshop idea and discuss the outcome and observations from its execution. Overall, our study demonstrates the potential of such a workshop design as a practical tool for teaching CT concepts to children. Finally, we critically examine the challenges of this approach and highlight the importance of proper technical and educational prerequisites for a successful implementation.

Keywords: Computational Thinking · STEM · algorithmic thinking · multi-disciplinary teaching

1   Introduction

The demand for Computational Thinking (CT) is expected to grow significantly in the next decade, particularly as more industries and administrations adopt digital technologies [6,18]. CT is foundational for Computer Science (CS), and a critical skill required in numerous fields, including engineering, science, finance, and healthcare [3]. It is a method for problem-solving that involves breaking down complex problems into smaller, more manageable parts and using algorithms and logic to analyze and solve them [17].

In Germany, there is still a significant education gap regarding CT [11]. However, Germany's educational system has not yet enforced mandatory CS classes in schools on a national level [8,12]. As of 2023, only 5 out of 16 states have added CS to their curriculum [13].

A major challenge is the shortage of CS teachers in Germany [13]. Many schools struggle to find qualified teachers to deliver effective and engaging CS classes [7]. This shortage of teachers can be a significant barrier to implementing mandatory CS classes across the board.

Meanwhile, as part of project GeNIUS[1], we propose a multi-disciplinary approach to infuse CS topics into natural science subjects like biology, chemistry, and physics, as natural sciences often require sensory measurements and algorithmic thinking in practice, both essentially covered by CT. This multi-disciplinary approach will allow children to develop CT from an early age and provide a better understanding of how different fields interplay to solve one particular problem. The project aims to conceptualize school lessons that make use of computer science practices, especially programming, to experiment on a natural science topic. The focus lies on summarizing success conditions in a school setting rather than evaluating the effectiveness of these lessons.

The first lesson that we are working on is on the topic of "energy-efficient insulation materials in houses". The lesson employs programming a micro-controller to conduct an experiment that measures how fast the temperature inside an insulated "house" levels out with the room temperature after initial heating. Before the start of evaluating the lesson series in a school setting, we were invited to hold a workshop at a local primary school with students in the 4th grade (ages 9–10). This gave us the opportunity to evaluate our concept and test our hypothesis that it is possible to successfully teach CT skills in a natural science context.

In this report, we summarize our experience on integrating CS fundamentals into natural science lessons. We further discuss the challenges involved and propose solutions to overcome them.

2   Background

This section introduces Computational Thinking and examines its implications in the educational landscape. Also, it introduces the Calliope Mini, an educational micro-controller designed to simplify coding and technology for children.

---

[1] https://www.genius-schule.de/.

2.1   Computational Thinking (CT)

In the context of this work, "thinking like a computer scientist ... requires thinking at multiple levels of abstraction" [17], which is especially useful for natural science education to review a topic from different angles. Wing further argues that "Computational Thinking (CT) is a fundamental skill for everyone, not just for computer scientists" [17].

In other definitions, CT is often divided into different aspects, e.g., pattern recognition, abstraction, decomposition, and algorithms [4]. The CT part of our workshop consists of developing a suitable algorithm for achieving the goal of the experiment. Although in essence all four aspects are employed for this task, our focus naturally lies on algorithmic thinking.

2.2   Calliope Mini

The Calliope Mini is a micro-controller, similar to MicroBit, explicitly designed for educational purposes to introduce children to coding and digital technology [1]. With a variety of programmable features, the Calliope Mini is a valuable companion in modern, digitally-enhanced classes. It features a range of components, including a 5×5 red LED matrix, a temperature sensor, and a light sensor, among others. It can be programmed with block-based programming languages using free editors, such as Microsoft MakeCode [2]. For teaching purposes, we chose block-based programming because it is visual and novice-friendly [16]. Programs can be transferred to the device via Bluetooth. In contrast to other mini-coding devices, such as Arduino, employing different sensors does not require complex wiring. Furthermore, unlike Raspberry Pi, it cannot compile locally and expects already pre-compiled programs. For the purpose of our project, we opted out for using Calliope Mini, as its use has drastically increased in schools all over Germany. Our hope is that these concepts will become part of the curriculum in the future. Therefore, we focus on providing lesson ideas for the tools that are already present. However, we firmly believe that the concepts of the lessons can be executed with any other micro-controller.

3   Workshop Design

To bridge the gap between natural sciences and CS, we designed an experiment on efficient housing insulation using Calliope Mini. It teaches algorithmic skills alongside physics knowledge about heat dissipation and insulation materials. This workshop has been developed as part of a research project called GeNIUS, which aims to develop interdisciplinary cross-curricular STEM lessons. Drawing inspiration from the maker culture, it is intended to foster learning-by-doing, collaboration, hands-on skills, and to give insights into the functioning of technical devices and scientific phenomena. The workshop is divided into two days with four blocks of around 40 to 50 min each and breaks as shown in Fig. 1. In the following, we describe the insulation experiment and the plan for each day in detail.
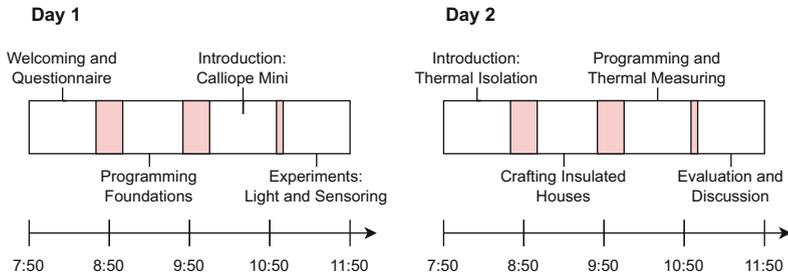
Fig. 1. Schedule for the two workshop days, including breaks (red). (Color figure online)

3.1   Main Experiment: Insulation in Houses

The selected experimental goal 'energy-efficient housing' for this workshop was chosen for its direct relevance and importance in the context of today's environmental challenges. Through the lens of insulated housing analysis, students were introduced to the effects of different insulation materials on a home's thermal stability, visualizing these impacts through data captured in real-time.

Similar natural science experiments are conducted throughout the school year by teachers in natural science classes. However, so far these have been done analogously - using analogous thermometers, monitoring and documenting the temperature by hand. We believe that this is exactly where CS can be integrated and allow the students to explore the possibilities of digital technologies.

The experiment is structured into four phases: insulating the house, implementing a measuring system, performing a measurement, and discussing the results. In the first phase, the students pick the material to insulate their houses. They then use the chosen material to insulate a cardboard box, which we use to simulate a house.

In the second phase, the students implement a program for Calliope Mini to continuously measure the current temperature and send it via Bluetooth to a mobile device. During the third phase, an external temperature sensor is connected to the micro-controller (see Fig. 2a). The sensor is put inside the insulated house, and the Calliope is turned on. Two heat pads are placed inside the house. The Calliope micro-controller is then connected via Bluetooth to an app called Phyphox, which receives the transmitted temperature values and automatically plots them in a diagram [14] (see Fig. 2b). The third phase takes around 30 min in total, where the students observe the temperature curve and compare it with the other insulated houses in the class.

3.2   Planning: Day One

We dedicate the first day exclusively to programming basics and introducing the students to the Calliope Mini and programming. Also, we set aside the first

(a) Assembling and Programming Calliope Mini

(b) Preparing House and Sensor Integration

(c) Experiment Start with Activated Heat Pads
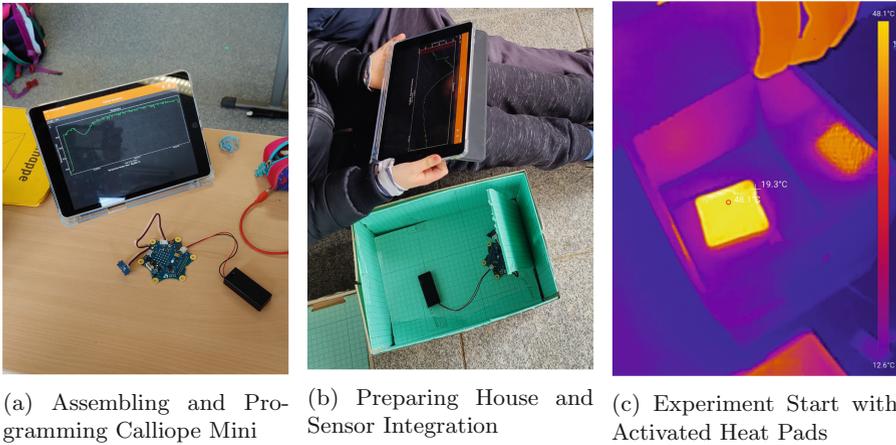
Fig. 2. Visual documentation of the thermal experiments in this workshop.

block for the pre-knowledge test. So the program for the first day is split into three blocks.

In the first block, we distribute hand-outs to the children with exercises to be solved individually using pen and paper. These provide the basic concepts used in (block-based) programming, including the definition of algorithms, conditions, branching, and different types of loops. In the last exercise, the children must construct an algorithm from scratch. In the last 10 min of the block, the students form teams of two, and each team receives a Calliope Mini. The teacher then leads a discussion about the different parts of the micro-controller.

In the second block, each team receives a worksheet with simple programming tasks to guide them through the different blocks. In the end, they are free to experiment on their own.

In the third block, the students learn to use embedded and external sensors and read their values. They learn how to send these values via Bluetooth and plot them on their tablets using the Phyphox app [14]. For that, they are given the task of writing a program that initializes the Bluetooth connection and sends light and temperature values via Bluetooth. They can experiment with the measurements, e.g., by covering the micro-controller with their hand or holding the temperature sensor in their hands, and see how the graph is affected.

3.3   Planning: Day Two

The second day was designed to further enrich the students' understanding of energy efficiency and insulation principles through hands-on experimentation.

In the first block, the session commences with a teacher-led discussion about energy efficiency and insulation, covering the fundamentals of heat-conductive and insulating materials. The very basics of heat-conductivity and insulation are presented by a short video discussing familiar objects from everyday life - e.g.

why do pans have a plastic handle. Energy efficiency is then presented from the housing point of view - the more energy the house loses, the more energy it needs to produce to stay warm. A thermal camera is used to visualize real-time temperature differences in the classroom, aiding in conceptual understanding. This is followed by a classroom discussion of the childrens' observations of housing insulation and what familiar materials have insulation properties.

During the second block, students are encouraged to tap into their creativity as they craft their insulated 'houses' from cardboard boxes and pre-selected insulating materials.

In the third block, students implement a program to capture temperature data using an external sensor and transmit the values to Phyphox via Bluetooth. Once the sensors are positioned and the Bluetooth connection is established, each team adds two heat pads to their 'house' and starts the measurement. During this period, the students are free to inspect the progress of other teams as the teacher guides a collaborative examination of the houses using the thermal camera. The experimental setup is left to operate over the final break.

The fourth block marks the end of the 30-min measurement period. It concludes with a teacher-led discussion of the resultant temperature graphs. Each team compares their data with others to ascertain the most effective insulation strategy, thereby reinforcing the learning objectives.

4   Results

In this section, we summarize our first-hand experiences gathered during the workshop's implementation. The assessment of a pre- and post-workshop survey is given in Appendix A.

4.1   Execution

During the execution, there were four instructors in the classroom, three of which were part of the design phase of the workshop and one a primary school teacher, who had no prior experience with CS, but whom the students were familiar with. For the execution, we used the Calliope Mini micro-controller and the external Grove temperature sensor. We provided two kinds of insulation materials: plain styrofoam boards and ones with aluminum covers on one side.

We carried out the experiment twice, with 20 students each. The two executions differed strongly due to the sudden misfortune of a failed router and, therefore, connection issues during the first two days.

Group 1 W/O Internet. The execution of Group 1's activities was heavily impacted by unexpected internet connectivity issues (see Fig. 3). The disconnection prompted the early implementation of all offline tasks, necessitating a reshuffling of the Day One and Day Two schedules. This disruption significantly
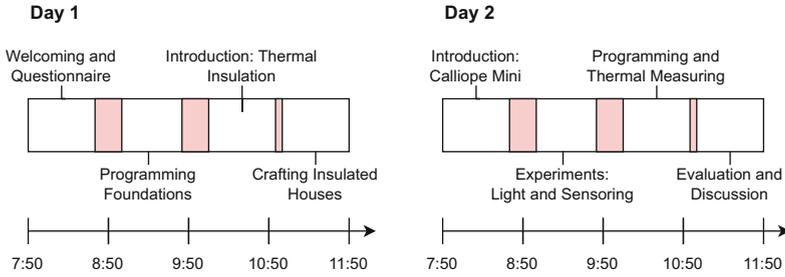
Fig. 3. The actual execution of Group 1, including breaks (red). (Color figure online)

dampened students' enthusiasm as they were unable to engage with the micro-controller as expected, leading to initial disinterest. Nonetheless, anticipation for the digital experiment built up by the day's end.

On the second day, the absence of an internet connection persisted, and we could not establish an offline server for Bluetooth implementation with the Calliope Mini. Our iPads proved futile in this offline environment, with mobile Internet hotspots providing only limited support. Resorting to our last available option, we manually compiled students' programs on a laptop and transmitted them via USB to the Calliope Mini. This workaround, although functional, resulted in queues and suppressed autonomous experimentation, fostering frustration.

While the experiment proceeded smoothly, students displayed waning concentration and interest, preferring play over participation in the post-experiment discussion. The feedback for this session indicated that it was "very boring," signifying the tiredness due to the unforeseen network outage.

Group 2 W/Internet. In contrast, the workshop's second iteration proceeded as initially planned (see Fig. 1). The students displayed high levels of engagement and enjoyment as they interacted with the micro-controllers and developed their programs.

During the outdoor experiment, with temperatures around $10°$, students were allowed to play outside. Nonetheless, they demonstrated keen interest, periodically checking their 'houses' and comparing their data with other groups. The thermal camera inspections of their houses added an extra layer of intrigue.

Post-experiment discussions witnessed active participation, with students enthusiastically presenting their findings. The successful execution of this workshop and the hands-on approach resulted in highly positive feedback, with the group endorsing it as "one of the best workshops they have attended".

5   Discussion

The workshop's successful execution demonstrated the feasible interplay of natural sciences and CT, an important step forward given the integral role these

systems play in our everyday lives. CT skills have become essential in the current era, cutting across numerous professional fields.

Our vision embraces a future where both educators and learners across all disciplines benefit from digital technologies and CT skills. Common perceptions, held by both young and old, frequently cast computers and software as enigmatic 'black boxes'-facilitating 'magic' in our daily applications. Our workshop's design aimed to demystify these notions for both students and teachers.

The participants were not only introduced to the formulation of simple, machine-understandable commands but also developed a clearer understanding of how devices interpret these commands. By leveraging digital technologies to convey insights derived from natural sciences-in this case, insulation and energy loss-we believe that the workshop fostered the development of multiple essential skills relevant to everyday life.

Creativity. During our workshop, we witnessed many out-of-the-box solutions. Not only that, but the children were eager to test out their programs in many scenarios, often proposing new ideas about where to measure temperature and what they would like to see next.

Digital Literacy and Electro-Mechanical Skills. The Calliope Mini has multiple ports. The children had to examine and conclude how to approach the wiring, i.e., which port does Calliope get power from, which port is used for data transmissions, and which for sensor measurements. Furthermore, they had to learn the Calliope Mini programming environment, which was a completely new application for them. Yet, they were quick to learn how to browse through the different block regions and find the blocks they needed.

Collaboration Skills. Throughout the workshop, the children worked on the tasks in teams of two. We observed great teamwork, where both kids would work together to achieve a common goal. For example, connecting the Calliope Mini with the tablet via Bluetooth is done by simultaneously pressing 3 buttons. Each team figured out that both groupmates were needed for this task. For the programming tasks, we observed different types of collaborations among the students. The first type would first discuss the solution and then implement it and test it together. The second type would involve one child implementing the program, another checking the provided material, and testing the program. They would then debug it together.

Graph Reading. Being able to deduce information from graphs is an important skill, which is learned at the end of 4th grade or the beginning of 5th grade in German schools [15]. Thus, the children for which the workshop was designed did not know how to create or read graphs. We had prepared material to slowly introduce them to this topic, introducing only the most necessary, like reading the maximum or the minimum, and interpreting steep and gradual lines. However, during the execution, we saw no need for this material as the kids were fascinated that they could see something moving and being created in real-time.

They were able to compare their graphs with the graphs of their classmates intuitively and to interpret what the highest temperature was and how fast it has been reached.

Further Aspects. Besides the skills mentioned so far, we also observed boosts in confidence and interest. Many children had expressed uncertainty before the workshop. After the workshop, all of them became more confident and eager to try out more.

Integrating such a high number of skills for achieving a task requires concentration and commitment from everyone involved. While this is a general challenge for children in this age group, we observed that in the second group even children who were known for disrupting lessons participated enthusiastically with good results.

5.1   Supporting and Hindering Influences on the Learning Process

In the following, we summarize the lessons learned from our workshops.

Didactical Aspects. Integrating active and collaborative teaching concepts makes the learning experience more engaging and interactive. In our experience, leaving time for semi-structured or self-guided exploration increased the students' confidence and interest. The ability to make mistakes without negative consequences encouraged the students to be curious and open-minded to new challenges and tasks of higher complexity. In our opinion, the school curriculum often fails to develop the skill of learning from one's own mistakes. We believe that CS allows for safe exploration and hands-on experimentation as programs can be arbitrarily often changed and adapted, and IDEs like the MakeCode editor prevent certain errors.

However, despite the many benefits of teaching with the Calliope Mini, it is essential to acknowledge that some teachers may face challenges when adopting new technologies and teaching methods. Therefore, teachers must have an open mind and be willing to learn and experiment with the device themselves [5,10]. Thus, delegating the implementation of this concept to teachers initially unfamiliar with Calliope Mini might result in sub-optimal teaching results. Only with proper support and resources, teachers can successfully integrate the Calliope Mini into their teaching practices and provide their students with a valuable and engaging learning experience.

Organizational Aspects. When teaching with the Calliope Mini, a frontal instruction approach is infeasible due to the technical nature of the device and the potential for individual issues to arise. As our and other collegial experiences show, having more than one instructor in the classroom is recommended to assist with troubleshooting and individual support.

While experiments in natural sciences already involve careful planning and preparation of material, this effort is increased with such a cross-curricular and

interdisciplinary lesson. However, with more frequent employment of the Calliope Mini platform, the initial overhead of acquainting the students with the programming environment, sensors, etc., is reduced and the system can be used productively after a very short amount of time.

Technical Setup. As mentioned in Sect. 4.1, the workshop was conducted twice. In the first iteration, the school's router broke on day one which rendered our technical setup unusable as the programming app on IOS requires server access for compiling the block-based programs to executables for the Calliope Mini. Therefore, the students could not work independently on the tasks which resulted in them not gaining a deeper understanding of the topic and unfortunately, losing interest. For us, however, it was a valuable experience from which we could deduce crucial technical conditions that need to be met in order for the lessons to succeed.

To effectively use the Calliope Mini, it is crucial to have a stable WiFi connection or a device that can compile programs. While it is possible to set up a local compilation server[2], it turned out that the programming app cannot be redirected to a different server. And though the IDE can also be used in the browser, it is not possible to upload the programs using Bluetooth as via the app. Instead, a USB wiring is required which requires in turn additional hardware and adapted worksheets with different instructions. Given the complexity of the workarounds, we recommend having an alternative ready in case of unforeseen network problems. The need for conservative approaches in a school setting is crucial for the successful integration of CS concepts in natural science topics.

5.2 Generalization of the Results

Stemming from our experience, we were able to derive the above-mentioned limitations and dependencies. Despite these, we believe that this workshop can be modified into a lesson series as part of the school curriculum for natural sciences or physics. In fact, this has been an ongoing work in GeNIUS project over the last months. The duration of the series depends on the pre-knowledge of CS concepts, especially block-based programming, and micro-controllers of the students. Furthermore, the age of the students is also a factor to be considered. We reckon that the older the students are, the less supervision and time they would need. Therefore, we believe that it is feasible for an experienced teacher to lead this experiment with up to 30 students of an appropriate age.

6 Conclusion and Outlook

In the rapidly digitalizing world, it is of utmost importance that children are introduced to Computational Thinking (CT) from an early age. Our work explores incorporating CT into a natural sciences lesson using the Calliope Mini.

---

[2] https://github.com/calliope-mini/pxt-calliope-static.

Through an experiential workshop, we observed the potential benefits of this approach. From fostering creativity and collaboration to improving digital literacy, and graph reading skills, the combination of CT with natural sciences opened up new avenues of learning. We believe our workshop plays a part in demystifying the notion of technology being an enigmatic black box, providing the first steps towards enabling children to comprehend and participate in the digital world around them.

It is crucial to acknowledge that the successful execution of the workshop hinges on the presence of technical and educational prerequisites. Stable internet access, availability of sufficient technical support, and teachers' openness to engaging with new technologies are all vital factors. Through this experience, we have found that even in failure, there are lessons to be learned and opportunities for refining our promising methodology.

We plan on exploring further possibilities to limit the dependencies on connectivity. As part of our project GeNIUS, we are working on modifying the workshop to be a lessons series, able to be conducted in schools solely by teachers. We plan on evaluating the success rate and further dependencies in German schools.

Our outlook is to conceptualize further lessons on other topics, taking into account also biology and chemistry. The main goal of the project is to evaluate the conditions that need to be met in order to successfully integrate CS concepts into natural science lessons. We furthermore plan on evaluating the learning effects of CT and of the respective natural science fields for each lesson series. This will be done on a wider test group, including both students and teachers. Furthermore, we are preparing a training specifically for natural science teachers, in order to show how CS concepts are an essential part of the natural sciences and, also, to teach them how to use micro-controllers. We hope that one day these lessons can be integrated as part of the natural science curriculum in schools.

We also hope that our insights will spur further research, propelling our education systems into a future where digital technologies and algorithmic thinking skills are seamlessly integrated.

Appendix A: Student Test

We conducted two tests with each student group, one administered in the morning before their respective workshop and a second one a week later. Their purpose was to assess the workshop's impact on the participants' knowledge and their preference to formulate algorithms in natural vs. semi-formal vs. graphical programming language.

Questionnaire. The questionnaire used in the tests was designed to cover two fundamentally different areas: First, understanding of the concept of an algo-

rithm, and second, the way in which algorithmic processes are described. Accordingly, the pre-test asked dichotomously whether the concept of an algorithm was known, followed by a multiple-choice task with four options asking which of the given everyday examples corresponds to an algorithm. In the second part of the questionnaire, the participants were asked to describe paths through simple labyrinths. The questionnaires pre- and post-test were structured identically. In the multiple-choice task, the answers offered were varied at the same level of difficulty. Similarly, slightly modified mazes were used for the open-ended descriptions of the paths through the labyrinths.

Evaluation. In the first question, the first item asked the students the dichotomous question "Do you know what an algorithm is?". Initially, a third of the participants (10 out of 30[3]) claimed familiarity with the term 'algorithm'. However, upon probing their comprehension with a follow-up multiple-choice item, only a single student could accurately identify the notion that best describes an algorithm in their everyday life. Post-workshop results showed a notable improvement, with 20% of students (6 out of 30) choosing the example matching the concept of an algorithm. Intriguingly, these correct responses were concentrated within Group 2[4] (representing 42% of its participants or 6 out of 14). Group 1, by contrast, produced no correct answers in the post-workshop survey. This stark difference in outcomes between the two groups underscores the significant role of technological stability in learning environments. We assume that the issues with internet connectivity for Group 1 of the workshop were likely the reason since coding and exploration of algorithmic structures were not possible. Thus, the results emphasize the impact of reliable technology in schools on the learning outcome.

In the second part of the questionnaire, the students were asked to describe a path through a labyrinth. In a sequence of items, they were told to describe the path

1. as they choose, without suggestions;
2. using abbreviations for up, down, left, and right;
3. using counting abbreviations (e.g., "repeat 2 times up").

For the final item, the labyrinth required a repeated sequence of steps ("repeat 2 times: up and left"). Here, the students were not given any suggestions regarding the description style. To evaluate these open-ended questions, we used the qualitative content analysis established by Mayring [9] using the following research questions:

How do students preferably describe the steps of an algorithm?
Are they able to adopt a semi-formal notation, and do they stick to it?

---

[3] We report here only on the participants who submitted both pre- and post-workshop questionnaires and who agreed together with their parents on participating in the survey.

[4] See Sect. 4.1 for a description of the two groups.

We defined the following categories:

Natural Language. Students use complete sentences in German;
Directions. Students use a sequence of instructions (e.g., "right, right, up");
Directions with Counting. Students use a sequence of instructions summarizing the same instructions(e.g., "2 × right, 1 × up");
Usage of Repeat. Students use the suggested repeat abstraction (e.g., "repeat 2 times: right, then up"), akin to the block-programming language introduced in the workshop.

In the first item of the question when no suggestion was made, 24 out of 30 students in the pre-test wrote their answers in natural language (post: 10). When asked to use abbreviations for the directions, 9 out of 30 students in the pre-test continued using full sentences mixed with the abbreviations (post: 6), 6 gave sequences of directions while 15 summarized by counting the same instructions (post: 8 vs. 14). Only 6 out of 30 students used the suggest repeat-abstraction in the pre-test (post: 12). For the final item, 3 out of 30 students used the repeated sequence (post: 6), all others used single instructions. Here, most students chose directions over natural language (pre: 18 vs. 5, post: 19 vs. 3). Notably, in Group 1 four students did not answer the question in the post-test while only two students in Group 2 dropped the question. As a trend, students were switching from the (more verbose) natural language abstractions to the shorter and more precise semi-formal descriptions when introduced to them. Their usage in programming tasks facilitates the adoption. Interestingly, in three cases, the usage of the semi-formal instructions improved the students' solution.

Limitations. Given the sample, its size, and the conditions of execution during the two iterations of the workshop, the conclusions and inferences drawn from it are hypothetical in nature and must be considered clearly preliminary. As the sample size increases, a new analysis will need to be conducted to see if any presumed efficacy and trends in response behavior (i.e., learning level) can be confirmed and generalized.

References

1. Abend, M., Gramowski, K., Pelz, L., Poloczek, B.: Coden mit dem Calliope mini. Programmieren in der Grundschule, Lehrermaterial (2017)
2. Ball, T., Chatra, A., de Halleux, P., Hodges, S., Moskal, M., Russell, J.: Microsoft makecode: embedded programming for education, in blocks and typescript. In: Proceedings of the 2019 ACM SIGPLAN Symposium on SPLASH-E, pp. 7–12 (2019)
3. Bocconi, S., et al.: Reviewing computational thinking in compulsory education: state of play and practices from computing education (2022)
4. Dong, Y., et al.: Prada: a practical model for integrating computational thinking in k-12 education. In: Proceedings of the 50th ACM Technical Symposium on Computer Science Education, pp. 906–912 (2019)

5. Hunt, N.P., Bohlin, R.M.: Teacher education students' attitudes toward using computers. J. Res. Comput. Educ. **25**(4), 487–497 (1993)
6. Kafai, Y.B., Burke, Q.: Connected Code: Why Children Need to Learn Programming. MIT Press, Cambridge (2014)
7. Klemm, K.: Lehrerinnen und lehrer der mint-fächer: Zur bedarfs-und angebotsentwicklung in den allgemein bildenden schulen der sekundarstufen i und ii am beispiel nordrhein-westfalens. Gutachten im Auftrag der Deutsche Telekom Stiftung, pp. 1–13 (2015)
8. Knobelsdorf, M., et al.: Computer science education in north-Rhine Westphalia, Germany-a case study. ACM Trans. Comput. Educ. (TOCE) **15**(2), 1–22 (2015)
9. Mayring, P.: Qualitative Content Analysis: Theoretical Foundation, Basic Procedures and Software Solution. Klagenfurt (2014)
10. Mozelius, P., Ulfenborg, M., Persson, N.: Teacher attitudes towards the integration of programming in middle school mathematics. In: INTED2019 Proceedings, pp. 701–706. IATED (2019)
11. OECD: OECD Employment Outlook 2019 (2019). https://doi.org/10.1787/9ee00155-en, www.oecd-ilibrary.org/content/publication/9ee00155-en
12. Sabitzer, B., Antonitsch, P.K., Pasterk, S.: Informatics concepts for primary education: preparing children for computational thinking. In: Proceedings of the 9th Workshop in Primary and Secondary Computing Education, pp. 108–111 (2014)
13. Schröuder, E., Suessenbach, F., Winde, M.: Informatikunterricht: Lückenhaft und unterbesetzt (2022). www.stifterverband.org/medien/informatikunterricht
14. Staacks, S., Hütz, S., Heinke, H., Stampfer, C.: Advanced tools for smartphone-based experiments: phyphox. Phys. Educ. **53**(4), 045009 (2018)
15. Ständige Konferenz der Kultusminister der Länder in der Bundesrepublik Deutschland: Bildungsstandards fädas Fach Mathematik Primarbereich (2022). https://www.kmk.org/fileadmin/Dateien/veroeffentlichungen_beschluesse/2022/2022_06_23-Bista-Primarbereich-Mathe.pdf
16. Weintrop, D.: Block-based programming in computer science education. Commun. ACM **62**(8), 22–25 (2019)
17. Wing, J.M.: Computational thinking. Commun. ACM **49**(3), 33–35 (2006)
18. World Economic Forum, V.: The future of jobs report 2020. Retrieved from Geneva (2020)

Author Index