tasks that slowly increase in complexity, reflect on results of previous tasks, generalize ideas, and with this support "learning by getting things to work."

References

Behringer R. (2013) Interoperability standards for MicroLearning. In: Proceedings of the MicroLearning conference. Stift Göttweig, Austria: 26–27 September 2013.

Bull P. H. & Patterson G. C. (2016) Strategies to promote pedagogical knowledge interplay with technology. In: Keengwe J. & Onchwari G. (eds.) Handbook of research on active learning and the flipped classroom model in the digital age. IGI Global, Hershey PA: 255–271.

Hromkovič J. (2018) Einfach Informatik – Daten darstellen, verschlüsseln, komprimieren. Klett und Balmer Verlag, Baar.

Hromkovič J. & Lacher R. (2019) Einfach Informatik – Lösungen finden. Klett und Balmer Verlag, Baar.

Papert S. (1980) Mindstorms: Children, computers, and powerful ideas. Basic Books, New York.

Papert S. (1999) Introduction: What is Logo? And who needs it? In: Logo philosophy and implementation, Logo Computer Systems, Highgate Springs VT: IV–XVI.

Juraj Hromkovič is a full professor in the Department of Computer Science at ETH Zurich. He is the chair of Information Technology and Education (ITE) and founder of the Center of CS Education (ABZ). He is a member of Academia Europaea and the Learning Society of the Slovak Academy of Sciences. In the last 15 years, he has been actively involved in the process of introducing computer science to the Swiss K-12 curriculum. He has written about 200 scientific articles and 25 books, some of them devoted to CS education.

Dennis Komm is a lecturer at the Pädagogische Hochschule Graubünden, ETH Zurich, and the University of Zurich. He is a member of the center of CS education (ABZ) and active in teacher education throughout Switzerland. He works in the area of algorithmics (with focus on online algorithms, randomized algorithms, and approximation algorithms) and CS education. He has published around 50 scientific articles and a textbook about online computation.

Received: 21 June 2019
Accepted: 4 July 2019

Towards a Creator Mindset for Computational Thinking: Reflections on Task-Cards

Mikko-Ville Apiola
University of Turku, Finland • mikko.apiola/at/utu.fi

> Abstract • Computational thinking (CT) skills are nowadays strongly advocated for educational institutions at all levels. CT refers broadly to skills of thinking about the world from a computational perspective, however, not necessarily referring to programming skills in particular. There is still a lack of consensus about what CT means, and how CT should be taught. This open peer commentary briefly discusses some ongoing trends of CT in response to the target article, which reports development, field testing and piloting of an extensive set of new learning materials for teaching CT. Recent calls for interdisciplinary technology education, creativity and open-ended problem solving in CT are highlighted.

Computational thinking

« 1 » Computerisation and digitalisation are changing the world in radical and unpredictable ways (Brynjolfsson & Mitchell 2017; Pajarinen & Rouvinen 2014; Frey & Osborne 2017). To meet future needs, more people who are skilled in the design of computational processes are required. Those design skills are nowadays commonly referred to as computational thinking (CT) skills (Wing 2008a; Tedre & Denning 2016). CT has become a popular concept and the teaching of CT is being widely advocated for educational institutions at all levels (Tedre & Denning 2016). However, as stressed by Matti Tedre and Peter Denning (ibid: 10), among other issues, it is still a challenge to define exactly what CT is, how it should be taught, and how teachers of CT should be trained. They also point out that practical challenges include undefined or obscure position of CT in curriculums, a lack of teacher training, and systemic problems in education.

« 2 » The target article by Valentina Dagienė, Gerald Futschek and Gabrielė Stupurienė contributes to the knowledge gap in understanding what CT is, and gives a concrete case-example of how CT can be taught. Their article reports the development, field testing and piloting of an extensive set of new learning materials for teaching CT in Lithuania. The research approach in their work is designing and evaluating learning materials, by using, as the authors call it, "a process-oriented approach" (§11). The authors link the learning process fostered by their new learning materials (task-cards) to creativity and constructionism. Around the globe, various initiatives are struggling to develop learning materials for CT. The materials and approaches presented by Dagienė and her colleagues are a much-needed addition. They provide a concrete way of inspiring both pupils and teachers, and an efficient way of bringing open-ended problem-solving into CT. The unplugged nature of the activities is also welcome. It has been argued that "Many of the core ideas of computer science are best introduced without machines" (Mike Fellows, cited in Bell, Rosamond & Casey 2012: 414).

« 3 » CT has its origins in the 1950s in the work of Donald Knuth, Alan Perlis, Edsger Dijkstra, and others (Tedre & Denning 2016). Seymour Papert's *Mindstorms* (Papert 1980) is one of the most ground-breaking and comprehensive contributions to CT in K-12 education. In its current form, CT was introduced by Jeannette Wing (2008a), who proposed that since computational processes are becoming important in all fields of life and sciences, learning CT would be beneficial for almost everyone. The "new CT" movement of Wing (2008a) started by mobilizing resources to bring CT into K-12 schools. CT, at the time, emphasized object-oriented thinking in software development and simulation, which was a rather narrow view (Denning 2017). Alfred Aho defines CT as the "thought processes involved in formulating problems so their solutions can be represented as computational steps and algorithms" (Aho 2012: 832).

« 4 » Linda Mannila et al. (2014) reviewed CT in K-9 education in Europe and the US. While the general tendency in many schools is still to emphasize the consumption of technologies, there is an increasing and expanding trend of teaching to understand and create, too. They identified ways that teachers in varying topics in

K-9 education are already involved in CT-like teaching, including activities such as analysing lyrics, composing problems, and programming robots. Shuchi Grover and Roy Pea (2013) reviewed computational thinking in K-12 education with various examples and ideas for practical teaching, including using makerspace ideology for the construction of tangible computational artefacts and informal hacker events, which are typical in the engineering tradition of computing. Judith Bell and Tim Bell (2018) show concrete examples of an interdisciplinary approach where CT is combined with music education at K-12 level. A number of progressive initiatives exist. However, in many cases, CT initiatives still emphasize the consumption of technologies or programming skills only, presenting a narrow view of CT (Tedre & Denning 2016). Dagienė, Futschek, and Stupurienė's approach, with a special focus on creativity and open-ended problems, is a highly welcome and valuable new addition to the pool of existing CT initiatives.

Creativity

« 5 » A recent report of the Board of Higher Education and Workforce of the US National Academies of Sciences, Engineering, and Medicine (NASEM) recommends that, especially in high-tech areas, learning outcomes should be associated with integrated education, such as critical thinking, communication, teamwork, and abilities for lifelong learning (Skorton & Bear 2018). Moreover, it is seen as crucial that STEM topics be integrated with humanities, arts, craft and design (HACD) topics (ibid). There are a number of important benefits of integrating HACD topics in STEM (Root-Bernstein 2018). In general, it is agreed by many that pedagogy should move from teaching a set of predetermined skills to teaching "the four Cs," critical thinking, communication, collaboration and creativity (e.g., Davidson 2017; Kivunja 2015).

« 6 » Also, research on experts in sciences and arts show that mastery is a function of time, intense focus, inspiration and deliberate practice applied to a particular field, not a function of genius, talent, intelligence, or any other fixed entity (Greene 2013; Ericsson & Ward 2007; Seligman 2011). In the domain of computational thinking, a *creator mindset*, a concept developed by Leong Hon Wai, as opposed to a *consumer mindset*, is a necessity in order to understand the fundamental impact that algorithms have on our lives, and how algorithms are designed. The creator and consumer mindsets are well aligned with the *dimensional mind*, including a combination of discipline and a childlike spirit, in contrast to a *conventional mind*, which is concerned with consuming information under pressure and in conforming to society. A dimensional mind transforms what it digests into something new and original, creating instead of consuming (Greene 2013: 176f).

« 7 » One example of creative pedagogies is Computer Science Unplugged (e.g., Bell, Rosamond & Casey 2012). It has been argued, in regard to mathematics and computing education, that "open unsolved problems are the creative drivers for mathematical activity, but children are 'taught' a version of mathematics based almost entirely on correct answers" (ibid: 404). These points are relevant in the teaching of computational thinking, as teaching of computing is sometimes implemented as well-defined programming tasks only, or the passive use of software applications (e.g., Tedre & Denning 2016). Therefore, the focus of Dagienė and her colleagues on creativity in their CT initiatives is needed.

« 8 » The authors and their research community have an extensive experience in creating task cards (§25). The task cards raise one question about the required time in solving the tasks. In the future, could the tasks, even more extensively than is reported in this work, be involved in large and small student projects where the open- or closed-ended aspects of projects could be varied? Thus, a question for the authors is: In the future, could the task cards act as components or "pointers" in computing projects, where cards could be linked together in larger stories, perhaps by using the methods of theatre, creative writing and storytelling? (Q1)

Three traditions

« 9 » A common approach is to divide CT into several components or skill areas. Valerie Shute, Chen Sun and Jodi Asbell-Clarke (2017) define components of CT as decomposition, abstraction, algorithmic thinking, iterations, debugging, and generalisation, which also seems to be the approach in the article by Dagienė and colleagues (§35). Wing (2008b) located CT in the intersection of mathematical thinking, engineering thinking, and scientific thinking. Engineering, mathematics and science are also the three traditions that constitute the basic pillars of computer science (Tedre & Sutinen 2008). It is good to recognize the separate traditions, cultures or tracks of computing. Peter Wegner (1970) argued that computer science is in part a scientific discipline, in part a mathematical discipline, and in part a technological discipline concerned in design and construction. This line of thinking is followed by Tedre and myself (2013) and Laura Benvenuti et al. (2018), who highlight the fundamental differences in aims, status of knowledge, and methodological views of the three traditions, and discuss the related implications for computing education. Recent research by Peter Larsson, myself and Mikko-Jussi Laakso (2019) has bridged the connections between the componential view of CT and the three-traditions view of computing with inspiring examples about the work of Alan Turing, on the processes of building the Electronic Numerical Integrator And Computer (ENIAC), and on the evolution of the physical symbol system, representing the differences in working approaches. Based on this line of thinking, the second question for the authors is: Could the three-traditions view of computing be used as one basis for building and classifying card-based open-ended learning CT tasks? (Q2)

Final thoughts

« 10 » The authors' work introduces the core concepts of computing in a playful way, teaching these and also the underlying thinking skills, but without suggesting any specific strategy, so that students have to create their own solutions (§65). The pedagogy in the authors' work also makes CT fun and inspiring, which is exactly what is needed. CT learning must be based on fun, mental freedom, and the possibility to explore. CT should not be about boring code commands, but about mental freedom, a passion for discovering, building inspiring new things, and changing the world. Thus, CT is about creativity and working with

open-ended problems. Even though computational processes may be run through computers, they are designed by and for humans. CT is not an isolated island, but mixes with practically all other domains of life including music, biology, dance, culture, mythology, and mathematics. It has been argued that "the beauty of computing is that it is engineering, science and art, a subject without clear borders" (Curzon et al. 2009: 201).

References

Aho A. V. (2012) Computation and computational thinking. The Computer Journal 55(7): 832–835.

Bell J. & Bell T. (2018) Integrating computational thinking with a music education context. Informatics in Education 17(2): 151–166.

Bell T., Rosamond F. & Casey N. (2012) Computer science unplugged and related projects in math and computer science popularization. Springer, Berlin: 398–456.

Benvenuti L., Barendsen E., van der Veer G. C. & Versendaal J. (2018) Understanding computing in a hybrid world: On the undergraduate curriculum front-end development. In: Proceedings of the 49th ACM Technical Symposium on Computer Science Education, SIGCSE '18. ACM, New York NY: 580–585.

Brynjolfsson E. & Mitchell T. (2017) What can machine learning do? Workforce implications. Science 358(6370): 1530–1534.

Curzon P., Peckham J., Taylor H., Settle A. & Roberts E. (2009) Computational thinking (CT) On weaving it in. In: Proceedings of the 14thp Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education, ITiCSE '09. ACM, New York NY: 201–202.

Davidson C. N. (2017) The new education: How to revolutionize the university to prepare students for a world in flux. Basic Books, New York.

Denning P. J. (2017) Remaining trouble spots with computational thinking. Communications of the ACM 60(6): 33–39.

Ericsson K. A. & Ward P. (2007) Capturing the naturally occurring superior performance of experts in the laboratory: Toward a science of expert and exceptional performance. Current Directions in Psychological Science 16(6): 346–350.

Frey C. B. & Osborne M. A. (2017) The future of employment: How susceptible are jobs to computerisation? Technological Forecasting and Social Change 114 (Supplement C): 254–280.

Greene R. (2013) Mastery. Penguin Books, London.

Grover S. & Pea R. (2013) Computational thinking in K–12: A review of the state of the field. Educational Researcher 42(1): 38–43.

Kivunja C. (2015) Teaching students to learn and to work well with 21st century skills: Unpacking the career and life skills domain of the new learning paradigm. International Journal of Higher Education 4(1): 1–11. https://files.eric.ed.gov/fulltext/EJ1060566.pdf

Larsson P., Apiola M. & Laakso M. -J. (2019) The uniqueness of computational thinking. In: Skala K. (ed.) Proceedings of 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO 2019). Croatian Society for Information and Communication Technology, Electronics and Microelectronics, Rijeka: 795–800. http://docs.mipro-proceedings.com/proceedings/mipro_2019_proceedings.pdf

Mannila L., Dagienė V., Demo B., Grgurina N., Mirolo C., Rolandsson L. & Settle A. (2014) Computational thinking in K-9 education. In: Proceedings of the Working Group Reports of the 2014 on Innovation Technology in Computer Science Education Conference, ITiCSE-WGR '14. ACM, New York NY: 1–29.

Pajarinen M. & Rouvinen P. (2014) Computerization threatens one third of Finnish employment. ETLA Brief 22. http://pub.etla.fi/ETLA-Muistio-Brief-22.pdf

Papert S. (1980) Mindstorms: Children, computers, and powerful ideas. Basic Books, New York.

Root-Bernstein R. (2018) STEMM education should get "HACD." Science 361(6397): 22–23.

Seligman M. E. P. (2011) Flourish: A visionary new understanding of happiness and well-being. ATRIA Paperback, New York NY.

Shute V. J., Sun C. & Asbell-Clarke J. (2017) Demystifying computational thinking. Educational Research Review 22: 142–158.

Skorton D. & Bear A. (2018) The integration of the humanities and arts with sciences, engineering, and medicine in higher education: Branches from the same tree. The National Academies Press, Washington DC.

Tedre M. & Apiola M. (2013) Three computing traditions in school computing education. In: Kadijevich D. M., Angeli C. & Schulte C. (eds.) Improving computer science education. Routledge, London: 100–116.

Tedre M. & Denning P. J. (2016) The long quest for computational thinking. In: Proceedings of the 16th Koli Calling International Conference on Computing Education Research, Koli Calling '16. ACM, New York NY: 120–129.

Tedre M. & Sutinen E. (2008) Three traditions of computing: What educators should know. Computer Science Education 18(3): 153–170.

Wegner P. (1970) Three computer cultures: Computer technology, computer mathematics, and computer science. Advances in Computers 10: 7–78.

Wing J. M. (2008a) Computational thinking and thinking about computing. Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 366(1881): 3717–3725.

Wing J. M. (2008b) Five deep questions in computing. Communications of the ACM 51(1): 58–60.

Mikko-Ville Apiola is a computer and educational scientist. He is currently working as a postdoctoral researcher at the Department of Future Technologies, University of Turku, Finland. In the past, he has worked in several Finnish and Tanzanian universities. His research interests include educational psychology, learning analytics, computing education, computational thinking, and ICT for development (ICT4d).

Received: 23 June 2019
Accepted: 30 June 2019