Chang, M. et al. (Eds.) (2019). Proceedings of the 27th International Conference on Computers in Education. Taiwan: Asia-Pacific Society for Computers in Education

Towards Computational Thinking in Scandinavia

Kasper KRISTENSEN
Maersk Mc Kinney Moller Institute,
Embodied Systems for Robotics and Learning,
University of Southern Denmark (SDU), Denmark
kakr@mmmi.sdu.dk

Abstract: This paper describes experimentations in the introduction of Computational Thinking in Denmark. An example being the new mandatory information technology course *Informatik* in the Danish high school system, which aims to strengthen digital literacy. Furthermore, we also describe how Computational Thinking is being taught in volunteer organizations. Given the need for recognizing typical patterns in the deployment and adaptation of Computational Thinking in national contexts, the author is currently engaged in a PhD project aiming at compiling a collection of best practices to teach programming, from Asia and Scandinavia.

Keywords: Computational thinking, pedagogy

1. Introduction

The world is getting more interconnected, more reliant on technology, and changing rapidly in response to disruptive innovations. We cannot with any confidence predict the future, but we can prepare the next generation. Computational Thinking (CT) has received global recognition as an essential skill for the twenty-first century (Wing 2006). Not least in Denmark (Caspersen & Nowack 2013) where a continued effort resulted in a new mandatory IT course for HHX (branch of high school focusing on commence and business) in 2016. Furthermore, there is a strong political interest in creating a similar course for elementary school students, a tendency we see globally (Resnick et al 2006). Therefore, proper dissemination of knowledge, and identification of pedagogical approaches for teaching CT is critical.

Within Denmark the desire to introduce IT to kids and young adults developed on two fronts, one focusing on school curricula and the other more informal and practice oriented. The paper looks at these 2 settings and how they both represent adaptation of CT to different contexts of learning, and finally how both can inform the newly developed CT courses for Danish primary schools.

2. Informatik

The subject matter of the new mandatory course *Informatik* centers around the "computational world" and it is a separated subject on its own (UVM 2019). Distinct from the normal use of digital tool in other courses. Indicating a major shift from the traditional approach which since the 1980s had been to integrate IT into other subjects (Caspersen & Nowack 2013). CT is not explicitly mentioned in the courses' study plan, which instead includes structural analysis, abstraction and decomposition, all fundamental parts of CT (Wing 2006). Compared to other courses within the Danish school system *Informatik* is unique, as it specifies which pedagogical approaches should be used in teaching its subject matter, not to the exclusion of all other approaches. But it is worth noting such decisions are normally left to the teachers' discretions in courses taught in Denmark.

Software and other tools used within the course are still left to the individual teacher, including choice of programming languages, a more typical balance of responsibilities in Danish schools.

Two central pedagogical concepts used in defining *Informatik* are *Worked Examples* and *Use-Modify-Create.*

2.1 Worked Examples

The purpose of a *Worked Example* is to show a solution to a problem, and to show the thought process behind the implementation (Atkinson et al. 2000, Caspersen & Nowack 2013). *Worked Examples* are often produced with multiple modalities in mind such as audio and visual. Often in the form video, allowing students to engage with the example at their own pace. By showing the process and considerations what goes into a design *Worked Examples* demystifies the process, an aspect which is inherently absent when only showing finished code or completed problem solutions. *Worked Examples* are especially useful at the early stage of skill acquisition, lowering the cognitive load by having theory and relevant examples parts saliently shown in the material (Atkinson et al. 2000).

2.2 Use-Modify-Create

Use-Modify-Create proposed by Lee et al. (2011), as a promising pattern for teaching CT, is a pedagogical framework where we allow the students to first engage with a given product. If the product is a game the students could play it, afterwards they should modify the code behind the game. This process could be informed by improvements the students wish to implement after the use phase, or small bugs they noticed in the code. As the students understanding of the code increases, they can move from the modification phase to the creation phase. First by implementing new features to existing products and later to create completely new products (UVM 2019).

3. Volunteer Organizations

There is another side to the introduction of Computer Science for young people in Denmark, which happens outside the classrooms. Denmark has a strong culture for volunteer organizations called *clubs*, where kids and young adults play games and engage in free-time activities, after school hours. And clubs focusing on new technology, such as 3D printers, micro-controllers and robotics, have appeared over the last couple of years. Limited research has been conducted in the area as such the following section is informed by informal participant observations and the author's experience as a board member and volunteer, teaching programming and organizing activities at a local *club*.

These organizations often admit children as young as 7 years old (Coding Pirates 2019), and gradually build their curriculum in tandem with the children's growing understanding and curiosity within the field. This means that the members could possibly start in a technology club around the second grade of elementary school and continue all the way through high school, around 17 years old, at which point they could start a degree in computer science or similar field. Afterwards some of these members can likely become volunteers themselves.

Interestingly, many volunteers working at these clubs have a background in teaching or working within the IT field, which often gives these clubs strong technical profiles in knowledge of general pedagogical approaches, theory and practical working knowledge. Therefore, these organizations could offer valuable insights for creating new elementary school courses. These volunteer organizations are working from the bottom up, where the ministry is working from the top-down to expose pupils to IT contents.

4. Discussion and conclusion

Even if both a mandatory course like *Informatik* and a volunteer organization offer courses to the same age group, they still have a very different profile. Generally, in the ladder the elective nature of the activity predicts higher perceived competence and Intrinsic motivation for the students (Ferrer-Caja & Weiss 2002). Because of these fundamental differences a pedagogical approach which is highly successful in one setting might still need adaption in another or might need to be discarded entirely. Therefore, it is very difficult to directly base a new CT course for elementary schools, merely on the

pedagogical patterns that exist in *Informatik* or those that are used in the *clubs*. More verification and observations are needed.

However, it is to be expected that simple methods like *Worked Examples* or *Use-Modify-Create* might emerge as pedagogical practices in other countries attempting the same implementation of CT in schools.

This paper lays the initial conditions and groundwork for a larger PhD project, which in the long term aims at compiling a comprehensive collection of the best practices and typical pedagogical patterns to teach programming, taking inspiration from similar attempts being implemented in various countries. Therefore, the main objectives for this project will be to observe and catalogue CT techniques in the classroom in Asia, discover best practices, and compare them to those in Denmark. Furthermore, there will be an intriguing opportunity to deploy qualitative methods in the Asian context, where quantitative approaches are typically considered essential for IT-related studies in education.

References

Atkinson, R. K., Derry, S. J., Renkl, A., & Wortham, D. (2000). Learning from examples: Instructional principles from the worked examples research. *Review of educational research*, *70*(2), 181-214.

Caspersen, M.E. & Nowack, P. (2013). Computational Thinking and Practice — A Generic Approach to Computing in Danish High Schools, *Proceedings of the 15th Australasian Computing Education Conference*, ACE 2013, Adelaide, South Australia, Australia, pp. 137-143.

Coding Pirates (2019, September) *Spørgsmål og svar*. Retrieved from https://codingpirates.dk/faq/

Ferrer-Caja, E., & Weiss, M. R. (2002). Cross-validation of a model of intrinsic motivation with students enrolled in high school elective courses. *Journal of Experimental Education*, 71, 41–68

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., ... & Werner, L. (2011). Computational thinking for youth in practice. *Acm Inroads*, *2*(1), 32-37.

Resnick, M., Maloney, J., Monroy-Hernandez, A., Rusk, N., Eastmond, E., Brennan, K., et al. (2009). Scratch: Programming for all. *Communications of the ACM*, 52(11), 60-67.

UVM (2019, Marts). *Informatik C, hhx, htx, stx, hf — Vejledning*. Retrieved from https://www.uvm.dk/gymnasiale-uddannelser/fag-og-laereplaner/laereplaner-2017/hhx-laereplaner-2017

Wing, J.M. (2006). Computational Thinking, *Communications of the ACM*, Vol. 49 (3), pp. 33-35.

WORKSHOP 2 - New Computer Simulations in STEM Education