The new computer science curriculum in Poland – challenges and solutions[1]

Maciej M. Sysło[1]

[1] Warsaw School of Computer Science, Warsaw, Poland
syslo@ii.uni.wroc.pl

Abstract. The new computer science curriculum has been introduced in Poland in 2017 (for primary schools) and in 2019 (for high schools). In this paper we first describe the building blocks of the curriculum and discuss the challenges we face. Then we focus on the curriculum for high schools and present how project based learning (PBL) supported by a flipped learning strategy can be used to organize content described in the curriculum and students' learning computer science integrated with other (school) subjects. Today when schools are closed and the near future is not certain, these two approaches become very important ways of teaching and even more important for students to learn at distance in a virtual environment.

Keywords. Computer science curriculum structure, curriculum implementation, flipped learning, project based learning (PBL), virtual environment

1 Introduction

The changes in the computer science education for all grades in K-12 have been the subject of nationwide proposals and discussions in Poland in 2014-2016[2]. Finally, the new national core curriculum for computer science (CS)[3] has been introduced to primary schools (K-8 grades; age 6-15) in September 2017 and to high schools and vocational schools (9-12/13 grades; age 15-19/20) in September 2019, after the curriculum has been approved by the Ministry of National Education in 2017 and 2018, respectively.

---

[1] Supported by a grant from Google.

[2] The new computer science curriculum benefits very much from our experience in teaching computer science in schools in Poland for more than 30 years (see [9] and [11]).

[3] In Poland, the subject is called informatics (pl. *informatyka*), however the term "informatics education" (pl. *edukacja informatyczna*) refers to any use of computers, informatics, and ICT in education (including ICT in other non-informatics subjects) as educational tools and methods. Computer science education (pl. *kształcenie informatyczne*) refers to rigorous learning and teaching computer science, it also contributes to general informatics education across other subjects as a tool and as an approach to problem solving.

Computer science is a compulsory subject in primary schools (1-8 grades) – at least 1 hour a week for 8 years, and in high schools (9-11 grades) – at least 1 hour a week for 3 years. Moreover, students in high schools (9-12 grades) may additionally choose computer science as one of the elective subjects taught according to the expanded curriculum for at least 2 hours a week for 4 years. The final examination (pl. *matura*) is also offered in computer science and recently it became quite popular among high school students.

In the European Union terminology (see [5]) computer science (informatics) education is included in the digital education which is to develop school learners' digital competences. There are three main ways to integrate this area within school curricula: (1) as a cross-curricular theme; (2) as a separate subject; (3) integrated into other subjects. The current core curriculum of the education system in Poland combines two approaches (2) and (1) with (2) – as a compulsory separate subject in K-12 and (1) – as a suggested cross-curricular and integrated approach in other, non-informatics subjects.

In the next section we present the computer science curriculum in more details and discuss the challenges which arise in its implementation and then in Section 3 we discuss our approach to teaching computer science all high school students, most of whom may not be interested in a career in IT.

2 The new computer science curriculum and its challenges

The curriculum structure

The curriculum consists of separate documents for each school level (grades 1-3, 4-6, 7-8, 9-11+ext.), however **Unified aims**, which define five knowledge areas in the form of general requirements, are the same in all these curricula. The most important are the first two aims and their order in each curricula[4]: (I) **Understanding and analysis** of problems based on logical and abstract thinking, algorithmic thinking, and information representations; (II) **Programming and problem solving by using computers** and other digital devices – designing algorithms and programs, organizing, searching and sharing information, using computer applications. The content of each aim, defined adequately to the school level, consists of detailed **Attainment targets**. Thus, learning objectives are defined that identify the specific computer science concepts and skills students should learn and achieve in a **spiral fashion** through the four levels of their education in K-12.

[4] The three remaining aims are:

III. Using computers, digital devices, and computer networks – principles of functioning of computers, digital devices, and computer networks, performing calculations and executing programs

IV. Developing social competences – communication and cooperation, in particular in virtual environments, project based learning, taking various roles in group projects

V. Observing law and security principles and regulations – respecting privacy of personal information, intellectual property, data security, netiquette, and social norms, positive and negative impact of technology on culture, social life and security.

According to Winch [14] (see also [13]), who identified the constraints for curriculum design, all three major types of knowledge: concepts, propositions and know-how should be introduced early in the curriculum, because these knowledge types are dependent on each other. As stated in [13], a promising approach to addressing this constraint is a spiral curriculum, such as that developed in Poland, where at each level unified aims are addressed but pedagogically the approach varies across three elements with the first element more important at lower levels and 2 and 3 become more important during progression:

1. problem situations, cooperative games, and puzzles that use concrete meaningful objects – discovering concepts and using heuristics;
2. computational thinking about the objects and concepts (see [2]) – developing algorithms, constructing solutions
3. programming – moving from a visual/block to text-based environment, including program analysis, its verification and debugging.

The idea of a spiral curriculum is based on Bruner's theory [3] in which manipulating real objects in earlier stages of cognitive development is important and later these may become more abstract representations. When the curriculum spirals upwards more complex concepts and methods can be introduced. Benefits of such a spiral curriculum include:

1) enhancement of key concepts and techniques each time the subject of computer science is revisited;
2) progression from simple concepts to more involved ones;
3) students can be encouraged to recapitulate their previous knowledge and apply to new problem situations.

This changing emphasis allows for some aspects of progression, critical for computer science, for instance: increasing difficulty of problems; enabling students to develop their problem-solving skills, moving from block/visual programming environments (as advised for K-6 grades) to a text-based environment (in 6-11 grades) such as C++ or Python (the most popular programming languages in high schools in Poland).

We are aware of obstacles and challenges and also open questions related to the implementation of the new computer science curriculum in schools in Poland. Although the spiral approach, on which design of our curriculum is based guarantees the successive development of computer science concepts, methods and know-how among students, we are not sure that this will sufficiently motivate and engage our students to learn, study, use, and develop their computer science knowledge, skills, and competencies of using technology through 11/12 years of schooling. In particular, although a rigorous computer science requires a solid background in mathematics and mathematical reasoning, we want to avoid what happens to mathematics education in schools which with years in school less and less interests students. We shall follow this aspect in Section 3.

The challenges

The delegates to the UNESCO/IFIP TC 3 Meeting at OCCE in Linz (June 2018) agreed on the 8 key challenges (see [4] and also [12]), previously identified by the task force EDUSummIT, that apply to all countries, however with varying importance at different stages in their processes of curriculum changes and implementation (see also [7] for some other questions related to rigorous computer science education in schools). There was also general agreement that the major challenges are:

Challenge 1: Lack of clear understanding of Computer science/Informatics as an academic discipline.

Challenge 7: Teacher professional development in a newly introduced Computer Science/Informatics subject is a challenge in quality and quantity.

Regarding Challenge 1, today in Poland computer science is widely recognized by the society, politicians, and decision makers as an academic discipline (in science and among technical disciplines); and as a result we observe that computer studies are the most popular among high school graduates (twice as many graduates choose computer oriented studies than the next popular subject – management).

Regarding Challenge 7, the lack of systematic teacher professional development in computer science, subject and pedagogical, is indeed the major challenge not only in Poland but all over the world. We work on improving the situation at university level for future teachers as well as offering systematic in-service training for working teachers. Recently, the proposal of the Council for Informatization of Education has been accepted and the Ministry of National Education will sponsor three types of in-service courses run by computer science departments at 7 universities: (1) for teachers of other subjects who will get a computer science teacher certificate (360 h); (2) for computer science teachers on implementation of the new computer science curriculum (120 h); (3) for teachers of elementary education (1-3 grades) who graduate from pedagogical faculties and are not prepared to teach computer science youngest kids.

The Ministry of National Education is also implementing a number of projects to enable teachers of other subjects to participate in various forms of further education for the improvement of their digital competences.

Let us also mention here:

Challenge 8: Identifying and allocating the additional resources for teaching Computer Science/Informatics.

This challenge is concerned with allocating digital resources to teach computer science and in some sense is closely related to Challenge 1. The education system in Poland was in some sense "ready" for the new computer science curriculum. In particular, (1) the subject has been taught in schools in Poland (under various names) for more than 30 years (see [11) and recently as an independent subject on each level of education in K-12; (2) therefore schools employ teachers who teach such subjects – they only need now an extra in-service training to meet the requirements of the new curriculum, especially on algorithmics and programming; (3) all schools have been equipped with basic hardware and software and are connected to the Internet; and (4) the most important and encouraging is the enthusiasm and readiness of school students on all education levels to learn how to program and use programming skills in various subjects and environments, such as robotics, games, computer science and ICT competitions, and to enhance competencies in computer science, and digital competencies in general.

In the next section we focus on two other Challenges 5 and 6 with regard particularly to computer science education in high schools in Poland. The remaining Challenges are discussed in [4], also with respect to Poland.

3 The new computer science curriculum in high schools

The following issues appear when implementing the new computer science curriculum in high schools, in particular in Poland, see [9, 10]:

1. A class in a high school may consist of students coming from various primary schools.
2. Challenge 5 based on [4]: The previous computer science curriculum has delivered poorly prepared students for computer science in higher (tertiary) education.
3. Challenge 6 based on [4]: Integrating computer science across other subjects in school is ineffective in practice.

We shall shortly comment on the first two issues and then present our approach to the third one.

Regarding the first issue above, computer science in high schools should formally be a spiral continuation of the subject under the same name taught in primary schools. However students coming to a particular class at a high school may come from various primary schools in which computer science lessons covered the curriculum not completely. We observe that some of the curriculum topics (especially those on algorithmics and programming) are not touched mainly due to the lack of subject knowledge and pedagogical preparation of teachers in primary schools,. In such cases, computer science teachers in high schools have to somehow repair a “broken spiral” of knowledge development by, for instance, special exercises and tasks which will connect the gap between the knowledge acquired in primary schools and expected level of knowledge in the area of broken spiral. We provide teachers with a variety of such tasks and methodological advices how to handle such cases. In our guide book for teachers, a topic from the high school computer science curriculum is augmented by adding tasks which play a role of introduction to this topic (a kind of warm-up) and are supposed to be covered, according to the curriculum, in primary school. Such tasks play also a role of reminder for students who met this topic in primary school.

We hope that the situation will improve regarding the first issue when a computer science examination (elective in the beginning) will be introduced at the end of primary schools. A proposal of such an exam has been submitted to the Ministry of National Education by the Council for Informatization of Education.

The second issue is much more complex to be easily solved by schools and tertiary institutions. As mentioned earlier, computer science studies at tertiary education institutions are very popular among high school graduates in Poland. Last year more than 40 thousands graduates applied for such institutions, however only 7 thousands of them took the final high school examination (pl. *matura*) in computer science. This may be interpreted that more than 30 thousands of graduates chose a career in computer science/IT not thinking about it earlier (in primary and high school) and not verifying their preparation for such study by taking the *matura* exam. To solve this issue we have proposed to tertiary education institutions to introduce an entrance examination in computer science. However such institutions are fully independent, especially in the private sector, and their main goal is to have as many students as possible regardless of the level of their preparation to study in computer science related areas. A much better situation is with graduates from vocational schools in the area of technology.

Finally we address the third issue, Challenge 6 from [4] and propose a solution which is now implemented in high schools in Poland.

The situation described in Challenge 6 as we observe in Poland is mainly due to: (1) the lack of integration of computer science (not ICT) with other subjects in the subjects' curricula and (2) the lack of basic computer science (not ICT) knowledge among teachers of other subjects. Unfortunately (see [8]), in Poland there has been no positive change in these two factors for last (20) years and we are very pessimistic about any change which might happen in the near future. Therefore, we have decided not to wait for any change in other subjects and their teachers and we propose the integration of other subjects with computer science within computer science lessons.

We implement and propose to use **project based learning** (**PBL**) as the main approach to organize content (to meet the computer science curriculum requirements) and to organize students' learning. PBL provides authentic, real-world contexts for learning, allows students to discover and recognize connections between various subjects and application areas, and also supports personal learning by leaving to students the freedom to choose projects and the way they explore various topics. Students, individually or collaborating in a group, are also more engaged to learn when they get a challenging project, as opposed to working chapter by chapter of a textbook. In the implementation of this approach we advise teachers and students that the most convenient and effective way to meet the expected outcomes is to use also a **flipped learning** methodology (see [1], [6]) – in the classroom students learn what the projects are and what they have to accomplish and most of the work they do off the classroom as a kind of homework, collaborating with other students, communicating with teachers of computer science and other subjects and getting their advices.

The computer environment as an independent platform has been designed to accommodate all educational materials, working procedures for teachers and students, and communication channels needed to organize the work of teachers and students according to the PBL strategy. There is also a place for collecting completed projects. The platform consists of three main sections:

1. Project proposals
2. Working space
3. Repository of completed projects

The platform will be open to all teachers and schools, however the main focus will be put on projects in computer science education in high schools and in vocational schools. A teacher in discussion with students chooses a project (in Section 1) for them, then students work on the project (in Section 2), and after they complete the project, its version, approved by the teacher, is put into the repository (Section 3). We now briefly comment on each section.

Ad 1. This section contains project proposals presented in a unified form. A project description contains: its goals (themes) with regard to computer science and other school subjects, motivating arguments (why the project is interesting and important), expected project outcomes (results), number of students in a team, time windows for its realization, suggested road map of the project, sample materials needed to complete the project, methods of evaluation and assessments (own and by a teacher).

Themes of projects in this Section come from various school subjects. Some of them are proposed by the team of the curriculum authors. We are also open on proposals coming from teachers, especially those teaching other subjects. Students' proposals of projects are also very much welcome since such projects motivate and engage them most. The proposals cover mathematics (computers in doing mathematics), physics (simulations), chemistry (computer experiments), history (Internet search), geography (statistical analysis of data), literature (discussion with Umberto Eco), foreign languages (Google translator and AI), and also physical education (physics of sport).

Ad 2. This section of the platform helps students to organize their work on a project they chose. First they learn in details about the project and its goals, choose a leader, split among themselves what they have to do, decide about the time schedule of their activities, establish communication channels. Then the results of their activities have to be uploaded to the platform in this Section and they fill in final rubrics regarding their own evaluation of the project, what they have learnt, and how they grade their achievements. Final grades for the team and for its members come from the teacher.

Ad 3. The repository contains descriptions of the projects which have been completed by students so far. A project is presented in the repository in a compact form of its realizations and outcomes, however with no personal data of students and teachers. There is an open access to this repository and we strongly encourage teachers and students to visit the repository and use its projects as teaching materials on various topics in computer science and, more important, as examples of integrating computer science tools and methods in other school subjects and disciplines.

Detailed description of our PBL environment for integrating computer science across other subjects in school will be the subject of another paper and reports. Some sample projects will discussed in a conference presentation.

Conclusions

In the presented approach to learning computer science in high school by all students, also those who are not interested in IT career, students learn computer science by solving problems coming from various school subjects (disciplines) while, at the same time, they develop their knowledge in those subjects what contributes to the integration of computer science with the other subjects. This way, integrating computer science across other subjects in school is more effective due to students' engagement when they are learning computer science.

Presented environment for the PBL used by students in a flipped learning fashion, is a valuable proposal for implementing computer science (and also other subjects) curriculum and – more important – provides a virtual environment for students to learn at distance when schools are closed.

References

1. Bergman, J., Sams, A., (2012) Flip your classroom, ISTE, Eugene.
2. Brennan, K., Resnick, M., (2012) New framework for studing and assessing the development of computational thinking, AERA, Vancouver, Canada.
3. Bruner, J.S., (1960) The process of education. Harvard University Press.
4. Coding, Programming and the Changing Curriculum for Computing in Schools, Report of UNESCO/IFIP TC3 Meeting at OCCE, June 2018, Linz, Austria, February 2019; https://www.ifip-tc3.org/working-groups/task-force-curriculum/
5. European Commission/EACEA/Eurydice, 2019. Digital Education at School in Europe. Eurydice Report. Luxembourg: Publications Office of the European Union.
6. Flumerfelt, S., Green, G., (2013) Using Lean in the Flipped Classroom for At Risk Students, Educational Technology & Society, 16(1), 356-366.
7. Hubwieser, P., Armoni, M., Giannakos, M.N., (2015) How to implement rigorous computer science education in K-12 Schools? Some answers and many questions. ACM Transactions on Computing Education. 15(2), 5:1-5:12.
8. Kompetencje przyszłości w czasach cyfrowej dysrupcji. Studium wyzwań dla Polski w perspektywie roku 2030. K. Głob (ed.), Miasta w Internecie, Evidence Institute, Warszawa 2019.
9. Sysło, M.M., Kwiatkowska, A.B., (2015) Introducing a new computer science curriculum for all school levels in Poland. In: Brodnik, A., Vahrenhold, J. (eds.), ISSEP 2015, LNCS 9378, Springer, pp. 141-154.
10. Sysło, M.M., Kwiatkowska, A.B., (2013) Informatics for All High School Students. A Computational Thinking Approach, in: I. Diethelm and R.T. Mittermeir (eds.), ISSEP'2013, LNCS 7780, Springer, pp. 43-56.
11. Sysło, M.M., (2014) The First 25 years of computers in education in Poland: 1965 – 1990, in: A. Tatnall and B. Davey (eds.): History of Computers in Education, IFIP AICT 424, pp. 266–290.
12. Webb, M., N. Davis, T. Bell, Y. J. Katz, N. Reynolds, D. P. Chambers and M. M. Sysło (2017), Computer science in K-12 school curricula of the 2lst century: Why, what and when?, Education and Information Technologies 22(2): pp. 445-468
13. Webb, M., Bell, T, Davis, N., Katz, Y.J., Reynolds4, N., Chambers D. P., Sysło, M. M., Fluck, A., Cox, M., Angeli, C., Malyn-Smith, J, Voogt, J., Zagami, J., Micheuz, P., Chtouki, Y., Mori, N., (2017) Computer Science in the School Curriculum: Issues and Challenges, in: A. Tatnall and M. Webb (Eds.): WCCE 2017, IFIP AICT 515, pp. 421–431.
14. Winch, C. (2013). Curriculum Design and Epistemic Ascent. Journal of Philosophy of Education, 47(1), pp. 128-146.