Computer Science Curriculum Proposal for Czech Grammar Schools

Daniel Lessner

Department of Software and Computer Science Education, Faculty of Mathematics and Physics,
Charles University in Prague, Czech Republic
lessner@ksvi.mff.cuni.cz

Abstract. *Computer science (CS) is not being taught systematically on Czech grammar schools. We believe we are missing here a fruitful opportunity to help to achieve secondary education goals. That is why we intend to create the necessary means and introduce CS on grammar schools. One of these means is a basic CS course programme, which covers what we think any grammar school student should know. In this contribution we explain why we believe CS belongs to grammar school and then we describe the proposed programme. Main focus is on its structure and content, along with the most important outcomes.*

1 Introduction

On Czech grammar schools[1], computer science (CS) is not a standard part of education. There is a compulsory subject about how to use digital technology, but no theory on efficient information processing. We give a more detailed description of the situation below, in this introduction.

We consider absence of CS a mistake. We believe that in an imaginary ideal case, CS is a regular school subject, as physics or history. Reasons for this make the last part of this introduction. In an effort to move towards the above mentioned utopia, we have developed a programme for a basic CS course. Its description is the core of this contribution, including the general goals of the proposed programme, its organization and structure. An individual section deals with the course content. Before we get there, the context in which we work has to be made clear.

1.1 Recent situation

Computer science (CS) is not being taught systematically on Czech grammar schools. The main reason is historical. The content of secondary education has not gone through any deep enough change for decades, therefore there was no formal need to teach CS. The following reasons then form a cycle: With no formal need, it is useless to prepare CS teachers. Without them, it is unrealistic to introduce CS into reworked curricular documents.

The most influencing phenomenon on grammar schools these days is the curricular reform. The former curriculum has been disassembled and put back together into a qualitatively different new system. Among other changes, the system of school-specific programmes and the notion of key competences are perhaps of the highest importance for us and we introduce them here briefly. More details can be found in [9].

Educational programmes are of two levels. Each school has to create its own unique school programme based on the *Framework Education Programme for Secondary General Education (Grammar Schools)* ([17], abbreviated as FEP). The main advantage of this approach is exceptional autonomy and freedom, possibly helpful also for changes like CS introduction. It can happen naturally based on each school individual needs, instead of any kind of directive state regulation.

All the content specified in FEP is obligatory for every school, hence it is important to examine how computer related education is included and organized there. The first part to look into is a subject called Informatics and ICT. It aims to develop the capability to use digital technology. CS is mentioned once and very generally. However, in the detailed characteristic of the subject we find dealing with information, algorithmization and introduction to programming included.

In other subjects we find many possible connections with CS. Obviously, we can use much of mathematics. In biology we have nucleic acids, genetics and ethology. Another area dealing with information processing is

[1] We use the term *grammar school* in the meaning of general secondary education, i.e. 15–19 years old students.

learning itself. The last connection with CS we mention here is the area of first aid and surviving emergency states. Efficient work with information and algorithmic approach are literally life saving here.

Finally, let us have a deeper look on key competences, the major goal shift we are experiencing. They are defined in FEP as a set of knowledge, skills, stances and values, structured in six overlapping units: Student has to be able to solve problems, to communicate, to study, to live with himself and with others, to be a good citizen and a good entrepreneur.

Teachers task is not to merely provide knowledge any more, they are to help students to develop their key competences. Classical subjects are now means for achieving other goals. It is therefore important to discuss shortly the potential of CS education to key competencies development: From a certain point of view, we might say that a substantial part of them simply *is* computer science. If we define CS using expressions as solving problems, transmitting, storing and using information and doing so efficiently, then we indeed find a broad overlap with competences to solve problems, to communicate (i.e. to convey information), to learn (i.e. to store and use information). This is even more obvious in the official key competences handbook [18]. Hence we claim that CS is able to provide a very useful tool for developing these competences.

1.2 Why to teach Computer Science

We believe CS should be involved in general secondary education[2]. The main reasons follow here. They of course touch the purpose of education itself, which is why the following text may feel more of philosophy than science.

Compliance with current grammar school curriculum The first source to be investigated when tackling formal education is the curriculum. The most important document in our case is FEP [17]. It is based on the basic strategy for national education, known as White paper [14]. As we have shown above, FEP includes many connections and indications leading to a substantive conclusion: CS matches very well with the current conception of our grammar school curriculum, and moreover, it could help significantly to achieve therein defined goals.

Real life usefulness Common every day situations may be understood and solved better with basic CS skills. The reason is simply that a large part of human life is about processing information, and in our competitive world, the more efficient is the better. Shopping in a supermarket can serve as an example of useful CS application in daily life. Will the shopping list lead to wander through the mall a few times, or will the shopping be done in a single pass? Which cash register shall one pick to get through fast? And obviously: how to fit all the goods into the backpack? The point is not to make people perform extensive CS based researches regularly. The improvement is based on the capability of doing so, along with confidence in doing so correctly.

Important field of study and its results Computer science is not a matter of temporary fashion, it is a well based science with important results and applications [5]. It influences virtually every area of human lives – including grammar school students, while these are not aware of its existence. CS importance itself makes a rightful place among other sciences in secondary education.

Reduction of mathematical education In Czech Republic, mathematics experiences reduction of both teaching time and content and decrease of its significance. One can argue that CS is just a field of mathematics. P *vs.* NP illustrates this well. It is one of the crucial CS questions, yet included among Clay institute millenium prize problems [3]. Studying CS naturally helps to evolve mathematical literacy [19]. Hence introducing CS into grammar schools shall indirectly help the unfortunate state of mathematics.

Information society One of the preferred routes of our development is to the information society ([14], [4]). It seems difficult to achieve this goal with general population unable to say at least what information is. Along with this basic knowledge, the ability of efficient information processing is essential. Teaching CS on grammar schools (and preferably also elementary schools) is a way to arrange this.

Future computer scientists According to [17], one of the goals of general secondary education is preparation for university studies. Many university students of computer related fields run into unnecessary trouble and disappointment, simply because they have no idea of what CS is about. We believe that tasting it (as well as other fields) in advance would help them to choose the right specialization.

[2] This is of course to be investigated – and the best way for doing so is to try to teach CS to grammar school students.

Beauty of the field Even on the very basic level, one can find beautiful and elegant ideas, worth to be taught just for pleasing and challenging the mind. Binary search serves as a good example. A brilliant idea, yet very comprehensible. Moreover, it is very useful for introduction of many substantial terms of CS, including complexity and recursion.

Someone could also argue that we shall start teach CS on grammar schools because abroad they already do it. We will name and reference only Israel ([6], [7]), Netherlands [13], USA [15] and Russia [11]. We do not consider this a reason for teaching CS. The point in worthy inspiration and experience provided by these countries.

We have gathered here strong reasons for teaching CS population-wide. The most important would be that CS has the potential to support global goals of general secondary education coming true. Further, CS itself is an important field of study and may be found very useful in everyday decisions as well as in professional life.

2 Course programme structure and organization

In this section we describe the course as a whole. We will first discuss the goals of CS education on secondary level and explain our choices. Knowing the situation and goals, we may proceed do decide how to organize the education and how to structure the content into individual topics.

The proposed programme is meant for one school year, 90 minutes weekly. This time frame is intentionally more than most of the school can afford nowadays. With unique school programmes and conditions, the optimal choice can hardly be found. We decided to reduce something bigger rather than inflate something possibly insufficient. With a year long programme we also deny the argument that there is not enough to say about CS on grammar school.

The programme is structured into modules, according to months. Their order is loose, many of them can stand alone. If the course can't be taught as a whole, the teacher is not forced to hand pick topics and make them match together, complete modules might still be used. Also, if some module is not useful for the students (e.g. in case when graphs are included in mathematics), it can be replaced with another. Thus, specific educational needs of each group may be met more easily.

Our CS programme is not to interfere with the usual and fairly estabilished ICT literacy education in any way. We welcome cooperation and are going to investigate a few possibilities. But with different goals, different subjects and their names are needed. That is why we do not intend to merge CS and ICT.

2.1 Goals

General goals of teaching CS are derived directly from the motivation described in the introduction and from relevant curricular documents, particularly [14] and [17]. In other words, CS education serves not only CS itself – we have to keep on mind a wider context too and respect the global goal of general secondary education. More specifically, CS education shall serve to develop key competences. This is also one of the factors when choosing topics to teach. Moreover, key competences remind us not to forget the important stances which might be taught during CS lessons (e.g. to leave repetitive work to machines, to think before acting, to work systematically etc.).

The following goals are already specific to CS: As with other sciences, the aim of their study on grammar school is to get familiar with their subjects, methods, connections to other areas, fundamental results and their applications. Students shall also be able to actually use the basics when needed. This is of course about finding the balance between many contradictory aspects, namely the depth vs. width and practical usability vs. theoretical importance of discussed topics.

2.2 Topics to teach

Although CS is rather a new field of study compared to traditional grammar school subjects, it is of vast broadness. In the variety of possible topics we needed to pick those which worth the most.

As we want students to get some overall idea what CS is about, we tried to find the key terms and principles. They should of course be of high importance, promising with interesting content, and timeless. We have chosen four such keywords. They shall be presented as laying in the core of computer science. Every lesson is, directly

or not, linked with them. Moreover, they are naturally related to each other. Other important notions emerge from these four, however, to keep things clear, we decided to suffice rather with less. The chosen keywords are *problem*, *information*, *algorithm* and *efficiency*: The problem is what makes us interested in CS at first. Information is what we need for finding a solution, preferably an algorithm – for the sake of efficiency.

The reader will find out soon, that many essential (compare e.g. [1], [16]) areas of computer science are omitted in this programme. This is caused by the time frame. If we want the students to understand what they learn, we have to provide them with adequate time. Having more of it, we would be glad to include also modules on automata, languages, data structures, sorting, computational graphics, cryptography or, of course, programming. These topics also will form the alternative modules.

2.3 Required level of understanding

First of all, the depth of understanding is yet to be found out. As there is not enough experience with exposing grammar school students to CS topics, it is not possible to determine the depth of their study reliably. Anything proposed in modules description is merely an estimate. Still, they are the target levels for piloting the course.

Generally speaking, we want students to understand CS not necessarily by formal definitions, but by the meaning and applications. We for example do not need to clutter the notion of algorithm with any computational model. Yet, the student shall still be able to explain why there will hardly ever be an algorithm to calculate π.

Another example is the P vs. NP topic. We consider it useful, even though there is no chance to introduce it in any detailed way on grammar schools. Still, students shall be aware of the existence of a distinction between solvable and fundamentally more complex problems. They shall also understand the rough, yet useful interpretation: when they are about to test exponential number of possibilities, they should rather think again. To achieve such a basic level of understanding, we suffice without nondeterminism and other advanced ideas. This approach leads of course to many simplifications and it is a matter of argument (and further research), how far they shall get. Our claim is that if some knowledge may provide some practical use, we should familiarize our students with it, even though they might not understand precisely the math behind.

3 Educational modules

In this section we describe the individual modules. In this limited range we decided to pick and comment only the most important and determining objectives, knowledge or skills instead of publishing detailed lists. Eight modules are shown below. The reason is that we expect the remaining two to differ significantly on each school. During the first month a revision of maths and logic is needed, as well as inspiration of students interest. The way seems to be obvious. There are plenty of interesting problems to solve available (and useful also for revising the necessary matters from other school subjects).

The second free slot is for the last month. It is meant for revising the important topics and stress their connections. After a year of study, both the teacher and students should know which modules they enjoyed as useful and interesting. The decision about the specific topic for the last month is to be based on this observation. Finally, we are prepared to proceed to describe the individual modules, which the proposed programme consists of.

Information The aim of this module is to let students understand the notion of information, which belongs to a group of terms like matter, energy, life etc., as explained in [4]. We feel we know what these are, yet are not able to define them acceptably. We want to use three definitions to help students to understand the term. One is based on a message with its interpretation [10], the other two on reduction of possibilities and uncertainty [12].

Students shall also be aware of the possibility and meaning of measuring information. If this is true, notions of redundancy and compression are only natural, considering also informational value of natural language. Obviously, binary system and combinatorics will needed here. Another related topic is classification tree. Students shall be able to compare the informational gain of different questions (e.g. smoking vs. diabetes vs. sex vs. age while deciding whether a heavily breathing person is under a heart attack risk) and to build the appropriate tree.

Graphs We would like to point out here that graph theory, even though widely used, is not being taught *anywhere* on grammar schools. We consider this a serious fault in the curriculum. Students, unaware of the well established science behind, keep thinking that drawings and schemes are inappropriate for actual intellectual work. During this module, they are to be shown many different sample applications to understand how powerful graphs are. Then they will learn a basic systematic search through a graph and a character based notation. This will also help them understand that the drawn shape is of no importance.

They will also find out, that although a drawing may be very helpful and provide invaluable insight, it may also deceive them and they shall rely only on the mathematics behind. After the module, students shall understand that there is a massive body of results to be used when dealing with graphs. They shall be able to recognize situations, when graph is a suitable tool, to use consistent terminology and to systematically search through a given graph.

Problem This module might not seem important, but our experience shows otherwise. If we want to introduce the concept of algorithm as a universal solution of a problem, we should deepen the intuitive understanding of the term problem. Students shall know how to define a problem, how a solution, how to choose between more possible solutions. The important concepts introduced in this module are decomposition into smaller or simpler new problems, state space, abstraction and black box.

Algorithm After seeing a few examples, it is time to introduce one of the essential concepts of CS, the algorithm. Students definitely have to understand the definition and know a few positive and negative samples. Along with algorithms, we want student to understand that what can be solved by an algorithm, shall be solved so, and people shall focus on what they are better at.

Efficiency It is somehow clear that there are more and less efficient ways to deal with a problem. This module will provide students with basic tools to tell them apart. We introduce the idea of a basic step and its counting, but will not proceed to the big O notation. Unlike in CS, in real life the constant matters sometimes and we do not want students to forget it. Binary search will serve as the main problem to work with.

After this module students shall understand that efficiency is not proportional to the length of algorithm description, that it matters especially with huge inputs, and that they are to think first before they act. They shall also understand that optimization cost something (at least the initial effort) and the relation between time consumed and perfection of the result.

Topological sorting This module is intended to be an opportunity to get deeper into all the basic notions and principles of CS taught so far while working on an application. We have decided for topological sorting of dependent tasks. Advanced students may add time constraints and search for the critical path (after understanding, what it is) or scheduling based on given resources. We have chosen topological sorting for the following reasons. Graph is used here in a non obvious way to work with dependencies. The argument for finding the algorithm is similar to the one used with eulerian paths, one of the motivational problem from the course beginning. Moreover, topological sorting and the consequent problems have their important applications in managing actual projects and scheduling. A good alternative to this module would be classical sorting,

CS and Humanities This module shows students the influence of computers and CS from a different angle. It is highly experimental and we would rather leave it for humanities classes. Unfortunately, their programmes are very conservative and do not reflect modern development. We believe students shall understand the world also in the future.

The first to tackle with is the Turing test. Students will familiarize themselves with the idea and with the fact that it is going to be practically passed. Therefore they should also make their minds about accepting it as a sufficient criterion for considering a system thinking, or not. The importance of this topic lays also outside CS, it helps people understand better what humanity is. A naturally related topic is singularity. Students shall be aware of the concept and have some basic idea of its possible consequences.

Having machines taking over more and more of our decisions, we have to ask where the responsibility for these lays. This drills down to questions of ethics, therefore free will and determinism (also related closely to information theory). Where is our free will, if any, coming from, and what prevents machines from exhibiting it? Students shall be aware of these questions and the main possible answers.

Advanced CS The last module is for introducing some advanced part of CS and let students to work on some larger CS related task. The choice of topics is wide, though we shall not forget there is almost no programming experience. Previous modules may be deepened, or we may choose something new, e.g. evolutionary algorithms. Another possibility is experimental complexity exploration on given algorithms. Comparing them, finding the best or worst input data or even proposing and testing enhancements. This module is followed with the last month, as described in the beginning of this section.

4 Summary

In this contribution we described the situation on Czech grammar schools regarding computer science and also explained why CS should have its place on them. To help making this true, we have developed a basic CS course programme. Its goals, structure and content constitute the core of this contribution.

Stated briefly, the goal is to improve students general capability to solve problems and communicate, and to familiarize them with computer science basic concepts, methods, objectives, findings and their applications. We have decided to achieve this using four keywords to build the course upon: information, problem, algorithm and efficiency. Another concept to teach is topological graph.

The course is to be piloted, evaluated and adjusted, then published along with the necessary supporting materials. The aim is that teachers, students, and parents would see the profits from CS education and include it into their school-specific programmes permanently.

References

1. Aho, A. V., Ullman, J. D.: Foundations of Computer Science, C Edition. Computer Science Press, New York 2000.
2. Bell, T., Witten, I. H., Fellows, M.: Computer Science Unplugged. Computer Science unplugged, Cantebury 2006.
3. Carlson, J., Jaffe, A., Wiles, A., eds: The Millennium Prize Problems. American Mathematical Society and Clay Mathematics Institute, Providence 2006.
4. Cejpek, J.: Informace, komunikace a myšlení. Úvod do informační vědy. Univerzita Karlova v Praze, Praha 2005.
5. Denning, P. J.: Computing is a natural science. Communications of the ACM. **50** 7 (2007) 13–18.
6. Gal-Ezer, J., Beeri, C., Harel, D., Yehudai, A.: A High-School Program. Computer Science. Computer, **28** 10 (1995) 73–80.
7. Gal-Ezer, J., Harel, D.: Curriculum and Course Syllabi for a High-School Program in Computer Science. Computer Science Education. **9** 2 (1999) 114–147.
8. Goode, J., Chapman, G.: Exploring Computer Science. Computer Science Equity Alliance, 2009.
9. Lessner, D.: Computer Science Education on High Schools. WDS'10 Proceedings of Contributed Papers: Part I - Mathematics and Computer Sciences. Prague, Matfyzpress. (2010) 110–115.
10. Lukasová, A.: Formální logika v umělé inteligenci. Computer Press, Brno 2003.
11. Marád, M.: Jednotné zkoušky z informatiky jako součást ruské maturity. Učitelský spomocník (ISSN 1214-9179), 21. 01. 2010.
12. Mařík, V., Štěpánková, O., Lažnský, J. a kol.: Umělá inteligence (4). Academia, Praha 2003.
13. Mašek, J.: První desetiletí informatiky v holandských středních školách, in Metodický portál, Články. 13. 11. 2009.
14. Národní program rozvoje vzdělanosti v České republice. MŠMT, Praha 2001.
15. Tucker, A.: A Model Curriculum for K-12 Computer Science: Final Report of the ACM K-12 Task Force Curriculum Committee. 2nd Edition. CSTA, 2006.
16. Vaníček, J., Papík, M., Pergl, R. a Vaníček, T.: Teoretické základy informatiky. Praha, Kernberg Publishing, 2007.
17. Rámcový vzdělávací program pro gymnázia. Výzkumný ústav pedagogický v Praze, Praha 2007.
18. Slejšková, E. (ed.): Klíčové kompetence na gymnáziu. Výzkumný ústav pedagogický v Praze, Praha 2008.
19. Faltýn, J., Nemčíková, K., Zelendová, E. (eds): Gramotnosti ve vzdělávání. Výzkumný ústav pedagogický v Praze, Praha 2010.