Olympiads in Informatics, 2019, Vol. 13, 171–192
© 2019 *IOI, Vilnius University*
DOI: 10.15388/ioi.2019.11

Computational Thinking Education through Creative Unplugged Activities

Michael WEIGEND[1], Jiří VANÍČEK[2], Zsuzsa PLUHÁR[3], Igor PESEK[4]

[1]*Holzkamp Gesamtschule Witten, University of Münster, Germany*
[2]*University of South Bohemia in České Budějovice, Faculty of Education, Czechia*
[3]*Eötvös Loránd University, Faculty of Informatics, Budapest, Hungary*
[4]*University of Maribor, Faculty of Natural Sciences and Mathematics, Slovenia*
e-mail: mw@creative-informatics.de, vanicek@pf.jcu.cz, pluharzs@caesar.elte.hu, igor.pesek@um.si

Abstract. Unplugged activities are well known in the computer science education. Creativity and computational thinking have been extensively researched and classified in last decade. In this paper we are focusing on creative unplugged activities and their potential in the classroom.

We propose a model consisting of four types of creative unplugged activities that are used in CS education and present the results of an international online survey in which 360 educators participated in 2018. The survey found out how far the model is supported by educators, the extent to which creative activities are used in the classroom, what intentions are being pursued and what educational potential is seen in the four types of activities. Based on results of the survey we present ideas and methods on how to include and integrate creative unplugged activities into CS education and some possibilities on how to change such tasks to be more creative.

Keywords: creativity, computer science, education, unplugged activities, computational thinking.

1. Introduction

A person is creative when she or he produces an idea or artefact that is new. Thus, creative persons change our culture by adding new elements. According to Czikszentmihalyi's model, creativity is not an individual ability but a system consisting of three parts:

- A *domain*, which is a system of symbols and rules (like computer science).
- A *field* that includes all persons that act as gatekeepers to the domain. They decide whether an idea or product is accepted as a new part of the domain.
- An individual person who is creative and creates new elements within the domain.

Margarete A. Boden (2007) distinguishes historical creativity (H-creativity) and psychological creativity (P-creativity). H-creativity takes place when people come up with ideas that are new and have never been shared in the history of mankind. This corresponds to Czikszentmihalyi's concept. On the other hand, P-creativity happens when a person comes up with an idea that is *subjectively new* to this person, but has been shared by someone else before.

Boden describes three types of P-creativity: combinational, exploratory, and transformational creativity. Combinational creativity involves the generation of unfamiliar (statistically unusual) combinations of familiar ideas. A typical example is a visual collage made of found photos. Visualisations of computer science (CS) concepts (using body language, Lego bricks, images) include combinational creativity. However, the CS concepts might be less familiar than the elements that are used for visualisation. Creative unplugged activities in the classroom clearly focus on combinational creativity. It is not forbidden that very bright high school students develop exploratory or even transformational creativity, but such events are beyond the scope of general education.

Czikszentmihalyi's theory is based on interviews with exceptionally creative individuals from science, art and business. For research on creativity in the classroom, we suggest considering local versions of global domains and fields.

1.1. *Creativity in the CS classroom*

The classroom situation can be considered as a microworld representing the society in a small scale. The parts of domains the students are familiar with are defined by the school curriculum, textbooks and learning materials on the internet. This environment can be called a "local domain". The local domain of computer science at an individual school reflects the global domain of computer science with one fundamental difference where the local domain might change rather quickly, whereas the global domain is rather stable and expands relatively slowly.

For example, with bright students and a supporting teacher in a CS class, in one year the local domain might expand quickly. Students can do unusual projects and share them with the local community. The students in the next generation might be less interested in CS, and the local CS domain shrinks again.

There is also a local field of CS at each high school teaching this subject, which mainly consists of CS teachers. They decide whether a student's work is "creative" in the meaning of "something surprising" or "unexpected". Students may also be members of the field, as they can be experts and have specific knowledge about the (local) domain of CS, and they can act as gatekeepers. The local field may be influenced by parents and students who are not in CS classes, and who are not part of the field because they are not familiar with the domain.

When creating a program, the programmer is not creative in the local domain of computer science unless she or he develops a new programming technique, which is not mentioned in the textbook or curriculum. However, she or he might have found this »new« technique on the internet. This would also be a creative act related to the local domain if this novelty is shared and accepted by the local field. Although the idea is not new within the global domain, it is a local novelty, and the local domain is extended by this idea.

Unplugged creative tasks might be creative if the local domain is extended by fresh ideas. For example, when students visualize a recursive algorithm in a role play, they do not create a new algorithm. The novelty is the new metaphorical representations for calling a function, passing parameters, processing data and so on.

Designed activities are important elements of CS classroom education. They consist of a task and some materials which are used to solve the task. “Unplugged” activities use tangible materials like blocks or any materials that are ad hoc available, pencil and paper or just own body, but not computers. The charm of “unplugged” activities is often the contrast between the materials and the CS concepts that are elaborated.

In research literature and educational programs, one can find a lot of advice on how to encourage and foster creativity. In this section, we will try to relate these pieces of advice to creative unplugged activities, considering:

- Activity design, including the task and the used materials. Combinational creativity can be encouraged by enlarging the variety of concepts in a person’s mind. The more concepts I know, the more “unusual combinations” I can find. Additionally, a person can practice finding new associations between concepts and learn to judge the value and novelty of ideas.
- Scaffolding during the performance of the activity. An atmosphere of strength and joy should be maintained during the performance of a creative activity. The learning environment should make it easy to focus and prevent distractions. An empty school yard might be a good environment for some creative activities. Classrooms should be designed to support the creative process. Scaffolding during the process should support and encourage the following operations (Czikszentmihalyi, 1996):
    - Clarify, analyse and re-define the problem or question to uncover new ways of looking at it.
    - Try to find connections between seemingly unrelated subject matters.
    - Challenge established wisdom by asking: how would I improve this?
    - Recognise alternative possibilities.
    - Look at things from different perspectives.
- Presentation of the results including rules for feedback. Students should not be discouraged by dismissing new and surprising ideas as mistakes. Creative people need self-confidence (Boden, 2007). The logical consequence is that encouraging a creative task should always lead to success. During the presentation, the field gets active. This is important for the creators since they must internalize the field, and they must learn to distinguish between good and bad ideas, new and old ideas. The presentation is a good opportunity to discuss novelty and values. Some ideas developed in creative unplugged tasks might get documented in some way, and they might extend the local domain of high school CS.

1.2. *Computational Thinking and Creativity*

In this section, we briefly describe the concept of computational thinking (CT) and its connection to creativity.

In 2017, Wing refined the definition of CT as " Computational thinking is the thought processes involved in formulating a problem and ex-pressing its solution(s) in such a way that a computer—human or machine—can effectively carry out.". Based on this, it can be concluded that CT involves three key components: algorithms, abstraction, and automation. Some researchers argue that it is still in an early stage of maturity with no solid definition (Lockwood and Mooney, 2017; Voogt *et al.,* 2015). In order to define and explain CT some researchers focus on CT skills (Curzon *et al.,* 2014; Atmatzidou and Demetriadis, 2016), others on its elements (Grover and Pea, 2013) where one of the elements is abstraction and pattern generalization (including modeling and simulation). Some researchers argue that CT is an activity, often associated with, but not limited to, problem solving (CSTA and ISTE, 2011, Beecher, 2017, p. 8, Haseski *et al.,* 2018).

CT can be seen as a fundamental skill for everyone, not just for computer scientists. It is applicable in either a computerised or unplugged problem-solving process. CT has the potential for application in a wide range of disciplines as the creative learning arrangements. Two main strategies are used for CT skills development: unplugged activities (activities that involve logic games, cards, puzzles, strings or physical movements to get in touch with computer science concepts such as algorithms, data transmission or data representation) and computerized activities (such as programming in arrow-based visual environments, programming in block-based visual environments, using textual programming languages, programming that is connected with the physical world) (Moreno-León *et al.,* 2018).

In this paper, we look only on creative aspect of CT. According to Korkmaz *et al.* (2017), CT is the extension of the problem solving skills of a person and the development of the creativity and critical thinking skills of the people by re-focusing. Thus creativity plays an important role in CT approach (Korkmaz, 2017).

CT fosters creativity in the classroom as it allows students to move from being technology consumers to technology developer. Creativity can also be augmented by CT (Mishra *et al.,* 2013).

In the field of CS, creativity usually aims at producing new techniques, both hardware and software, that can provide solutions to practical problems. However, creativity involves the same kinds of cognitive processes that generate answers in computing as well as in the natural sciences (Saunders and Thagard, 2005). Moreover, in CT, creativity could be seen as an ability of applying imagination to create a physical object or some mental or emotional construct (Korkmaz *et al.,* 2017) that is judged to be novel and also to be appropriate, useful, or valuable by a suitably knowledgeable social group (DeSchryver and Yadav, 2015).

1.3. *Unplugged Computer Science Activities in the Classroom*

Many important topics in informatics or in other areas of science, society and technology connected with computer science can be taught without using computers.

The CS unplugged activities provide a scaffolding for a constructivist approach to introducing topics in computer science, without the need to learn programming first (Bell, 2018). The connections and some ideas and working processes are often easier to explain with hand-made activities (Bell *et al.,* 2012). These activities can be simpler and can be performed without computer. They can be used to present the important parts (e.g. sub-knowledge) of a big system or a working process in a concept.

Unplugged CS activities support computational thinking, although for this to be effective they should be used in a context where they will be linked to implementation on a digital device (Bell, 2018). There is another important aspect: by using these tasks, they could help to "de-mythize" the computer in the eyes of children as a "magic box who knows and does a lot and we do not understand how".

In this section, we briefly summarize the advantages and disadvantages of using unplugged activities in the classroom.

1.4. *Advantages of Using Unplugged Activities in the Classroom*

Using unplugged activities helps to think not only about computers and computer science, but about interdisciplinary activities without school-subject boundaries. Students can see the relationship(s) between different subjects and disciplines better.

The activities are mostly combined with physical movements. The aims of theories about learning by doing and learning by moving promote breaking out of the daily routines and school-life. (Bell *et al.,* 2015; Rodriguez *et al.,* 2017; Thies and Wahrenhold, 2013)

The activities give the basis for cooperation and teamwork. They can support the development of communication as skill. Students need to speak about a problem: they need to formulate the problem, the solving strategy, and the solution.(Bell *et al.,* 2015; Feaster *et al.,* 2011) In some activities, they need to present artifacts to an audience and handle the comments.

1.5. *Disadvantages (Problems, Challenges) of Using Unplugged Activities*

The computer unplugged activities are not only about "playing". They need a strong basis: a stable preparation. Mostly they give a lot of work for teachers. Not only in preparation (initiative) processes, but they can be barrier for the teacher. They need to change their mind from "exact" computer science to non-exact open ended situations; which necessitates changing their teaching style. It is hard work to keep the focus (why we started) and to manage time, as well.

Students can refuse to do the activity – “playing” is not always a means of motivation. (Rodriguez *et al.,* 2017; Thies and Wahrenhold, 2013). We present some additional comments about the problems of using unplugged activities in chapter Research.

2. Definitions of Creative Unplugged Tasks

In this section, we present a model for categorizing different types of creative unplugged classroom activities on computational thinking. The model consists of four categories which are described and presented with an example each.

Category 1: Invent an algorithm

Invent an algorithm that solves a given task and present it without a computer.
Here, the creative challenge lies in devising and formulating an algorithm with appropriate commands and data representations.

Example: Invent with your team how to move a sequence of 0s and 1s in the playground from A to B across multiple stations. Only body language may be used.

Possible solution: Represent a zero by stretching out both hands and represent a one by a jump.

Category 2: Find an application

For a computer science algorithm or concept, find a new situation in which this algorithm or concept could be applied.
Here, the creative challenge lies in the design of a story. The algorithm is given. It needs to be understood so that you can check if the story fits as the application context.

Example: Describe a situation in which you can use the binary search. Show the situation in a small role play. Use things from this room.

Possible solution: Search for a word in a dictionary.

Category 3: Find an example

Find an algorithm that has certain given structural features (e.g., loops, recursion, function calls) and represent it in some way.
Here the creative challenge lies in finding both a task and a suitable algorithm. These kinds of creative tasks solve teachers e.g. when looking for illustrative examples of a specific programming technique.

Example: Here is the definition of a function with a parameter in Python notation.

```
def crumble (thing):
      Open the hand.
      Put the matter in the hand.
      while not thing has the form of a ball:
      Move the fingers of the hand.
      Give it back.

# Main program
crumble(sheet of paper)
```

Imagine a program in the style of the example and write it down. The program text should contain a function definition with at least two parameters. As parameters you use everyday objects from this room.

Possible solution: Wrap a thing in a foil.

Category 4: Find a visualization

Invent a visualization for an algorithm or concept of computer science.

Here the creative challenge lies in finding a representation that is understandable to the public. Various materials may be used, e.g. crayons and paper, role-playing with props, pantomime, Lego building blocks. These kinds of creative tasks solve e.g. book authors when looking for illustrations for a textbook.

Example: There are cards with terms from computer science, which should be visualized. Pupils draw a card and present the term through mime. The audience has to find out what it is.

Possible solution: Visualization of a WLAN access point through mime.

3. Research

Survey

In 2018, an international online survey was conducted. The questionnaire was translated to the national languages, and it was sent through different channels to computer science educators. In Table 1, the distribution of the educators based on their country of origin is presented, with the total number of answers, 360.

The respondents' pedagogical expertise is presented in Table 2. Most of them have a background in secondary education.

In the questionnaire, we first presented four different types of creative unplugged activities, and then asked the respondents how clear they thought the definitions were. The responses were given on a scale from unclear (1) to clear (5). Fig. 1 presents the results.

107 out of the 360 respondents (more than ¼) claimed that they know other creative CS activities without computer besides our four categories. 38 respondents gave one or more examples, some respondents wrote general comments.

Table 1

Respondents origin

Country of origin	Number of respondents
Czech republic	133
Lithuania	8
Slovenia	25
Germany	153
Japan	14
Other countries	27
Total	360

Table 2
Respondents' professional experience

Pedagogical expertise	Percentage
Teaching computer science-related content at a primary school	28 %
Teaching computer science related content at a secondary school	88 %
Teaching computer science related content at a university	14 %
Authoring or co-authoring a textbook in the field of computer science	12 %
Research in the field of computer science	13 %
Teacher training in the field of computer science	27 %

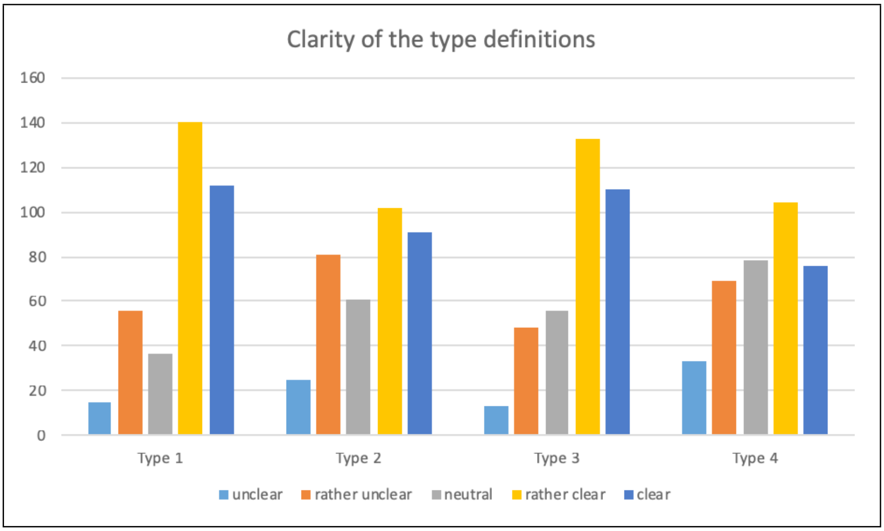


Fig. 1. Graph representing clarity of the categorization.

We can identify only about one third of these 38 responses as examples of creative unplugged activities, moreover, most of the presented examples were unplugged tasks but were not creative, as they were typical problem-solving activities (e.g. ordering data, comparison of algorithms). Some examples were not even from field of CS (e.g. painting). In some tasks, it was difficult to decide whether this task is creative or not, for example, the response „puzzles – sudoku" we could not decide whether the respondent meant solving sudoku or creating a new puzzle.

Some of the responses described concrete creative activities which were written very sketchy, yet we can say that they can be categorized to one of our four types e.g. „Representing the TCP/IP protocol stack through role play using envelopes as packets" which is Type 4, create a visualization.

We can say that only a few answers out of the 38 responses were creative unplugged activities that cannot be directly mapped to our model of activity types. These answers were not oriented to algorithms but to data transfer, data coding or finding a pattern. If we wanted to create one new type of activities based on the questionnaire responses, we could find it probably in „more complex tasks“, „systematisation“, „discussions about possible solving strategies“ or „demand of informatics in the society, economy“.

Next we asked the respondents about their experiences with creative tasks. On the question “How often have you spent the last 12 months doing creative tasks without a computer in computer science education or a computer science course?”, 50% of the respondents said they used them once or twice, 27% used it more than twice and 23% said they never done them in the classroom.

On the question “If you have engaged in creative tasks what was the reaction of the students?” we got the results presented in Table 3.

59,4% of all participants observed students’ positive reactions and 10.7% of all participants negative or no reactions.

Next we asked educators if they could tell us why and how they used creative unplugged activities. Respondents could check multiple options. The results are presented in Table 4.

We asked the educators about their opinion on the educational potential of creative tasks. More precisely, we asked them about specific potential and which type of creative tasks can be used instead. More answers could be selected at each question. Results for different purposes are in Table 5.

Research on the results of the educators with the positive students’ feedback

As the next step, we compared the attitudes of educators who reported positive and rather positive students’ reactions with those who got a non-positive reaction (neutral,

Table 3

Students reaction

Student’s reaction	Percentage
Positive	13.6 %
Rather positive	45.8 %
Neutral	12.7 %
Rather negative	6.6 %
Negative	0.5 %
No reaction	3.6 %
No answer	16.9 %

Table 4

Usage of creative unplugged activities by teachers

Intention	Percentage
As an activating entry into a new topic	51 %
As a suggestion to think about certain abstract concepts	47 %
To promote creativity	26 %
As training, e.g. to practice the correct use of technical terms	25 %
As a diagnostic tool to identify a prior understanding of a topic or misconceptions	16 %
As an entertainment element to relax a course	38 %

Table 5

Educational potential of 4 types of CS tasks.

Educational Potential	Type 1 *Create an algorithm*	Type 2 *Find an application*	Type 3 *Find an example*	Type 4 *Visualisation*
A replacement for programming tasks providing a comparable learning experience in less time	72 %	50 %	44 %	46 %
An enrichment to a lesson or lecture to make the content more relevant and attractive to the participants	68 %	58 %	39 %	62 %
As a encouragement for students to think about computer science concepts	61 %	67 %	54 %	60 %
As a help to develop transferable skills	62 %	71 %	49 %	56 %

Table 6

Attitudes of educators with different reaction

Intention	Positive reaction group (N = 214)	Non-positive reaction group (N = 85)	Chi-Square
As activating entry	62.1 %	57.6 %	.472
As a suggestion to think about certain abstract concepts	**61.2 %**	**42.4 %**	**.003**
To promote creativity	**20.0 %**	**35.5 %**	**.009**
As training, e.g. to practice the correct use of technical terms	30.6 %	29.4 %	.845
As a diagnostic tool to identify a prior understanding of a topic or misconceptions	15.3 %	20.6 %	.296
As an entertainment element to relax a course	36.5 %	48.1 %	.067

rather negative, negative). In the first group we have 214 respondents and in the second group, 85 respondents. We excluded the respondents (N = 64) who did not provide an answer.

We used Pearson's Chi-square test to check if there exists a statistically significant association between positive reactions of the group, and the intentions of usage in the classroom. Results are in Table 6.

We also checked whether there exist any differences in their opinions regarding the educational potential of the four types of unplugged creative activities. The results show that both groups share similar views ($p > 0.5$), except when activity is viewed as a encouragement for students to think about computer science concepts where $p = 0.05$ for type 3 and $p = 0.03$ for type 4.

Discussion

From Fig. 1 it can be observed that types 1 and 3 are clearer and better understandable for the educators than type 2 and 4. It's interesting because type 2 and type 3 are very similar. It seems that educators feel more confident with activities that ask for specific structural features than with more general concepts. One of the reasons is that activities and tasks that fit in type 3 category are more frequently used in CS teaching.

We can conclude from the answers presented in Table 4 that educators use creative unplugged activities as activating entries or as suggestions to think about some concepts, which are also the same reasons why they are used to promote unplugged activities. It's interesting to notice that more than one third of educators use unplugged activities also as an entertainment element in the classrooms.

As a replacement for programming tasks presented in Table 5 most educators think that activities that fit into Type 1 (Create an algorithm) are most useful and appropriate. The reason for such results might lie in the fact that Type 1 activities are conceptually very similar to the programming tasks, only the allowed tools are different.

For the other three types, it's not that easy to see how such activities can replace programming tasks. It's interesting to notice that educators don't feel that Type 3 activities can replace programming tasks, although such activities develop understanding of some concepts similar to programming tasks. Most of educators think that all types of creative tasks, except Type 3 tasks, are suitable for use in the classrooms as enrichment.

Analysing the answers in depth, we can see that almost 90% of respondents that checked Type 3 also checked tasks of Type 1 and Type 2. Educators think that tasks of Type 2 are most appropriate to be used as encouragement, although other types of activities are also used in classrooms. We can conclude that educators think that finding a new situation encourages students to think about CS and also helps students to use their CS knowledge in different areas of their life. Most educators think that tasks of Type 1 and 2 help students develop skills that can be used in other scientific fields. The reason might lie in the fact that educators think that the first two types help in developing computational thinking which can be later applied in various fields.

Educators who reported positive students' reactions on creative unplugged activities ("successful educators") tended to use these activities more frequently to encourage thinking about CS concepts and *less* to encourage creativity in general. This could be a hint that these educators were successful *because* they tried to encourage thinking about CS concepts. However the perception of students' reactions may be subjective.

The fact that almost 60% educators noticed positive reaction must be used to further encourage the use of unplugged activities in classrooms.

The questionnaire explained four types of creative unplugged tasks and gave examples. For many CS teachers, this might have been the first contact with unplugged activities and creativity in the classroom, and they might have had problems with understanding short introduction of the survey.

A possible reason that teachers do not use creative unplugged tasks in CS education so much might be that focusing on computational thinking is not established so well in CS education, and the fact that there aren't that many available prepared creative unplugged activities which educators could use. Another reason could be the perceived lack of time for these activities which seem to be time-consuming.

From curricula point of view, teachers saw application of these tasks as an introduction to understanding how computers work, in media education and painting art activities and in creating criteria by ordering data. This might indicate that some teachers connect creativity more to using ICT than to computer science.

The results of our survey show that the creative unplugged activities are not used often in education. Therefore, in next chapter we propose suggestions how to prepare and use creative unplugged activities to develop and encourage creativity in computer science education.

4. Suggestions for Creative Unplugged Activities in Computers Science Education

In this section, we present some ideas how to develop creative CS unplugged classroom activities fulfilling three conditions:

- They challenge creativity.
- They touch a computer science topic.
- They can be performed without a computer.

The first three subsections discuss how to create a new creative CS task, subsection 4.4. shows how to use the created activities in the learning process.

4.1. *How to Express Ideas*

In this section we focus on the potential and challenges of different "unplugged" expressive means: writing, drawing, concept cartoon, building with physical constructions bricks or found material and role plays. Generally, each type of creative activity can be performed using different expressive means.

Writing

The most obvious way to express an idea is just to write it down. The product could be just a list of words or a story with an action plot.
Example tasks:

- Write down the idea of an exciting programming project that is based on case distinctions, which could be coded by if-elif-else statements.
- Write a christmas story, in which an iteration takes place.

Drawing

Students may create drawings alone or collaboratively in a team. The advantage of collaborative drawing is that there is more communication. Two or more students sit at a table around a big sheet of paper. They can partly work independently. Each person might be responsible for a certain part of the image. Example tasks:

- Think of five or six substances you can find in this room (iron, wood, …). Create an image that illustrates how to distinguish these substances (decision tree). (Type 1)

- Draw a robot ship with sensors that can find and remove plastic garbage from the ground of the ocean. (Type 1)
- Many computer programs contain while-statements. In a while-statement a sequence of activities is executed again and again as long as a certain condition is true. Example: "While you are thirsty, put some water in a glass and drink it." In this case, the condition is "you are thirsty", and the sequence of activities is "put some water in a glass and drink it". Create a comic-book story that illustrates a while-statement taking place in everyday life or in a fantasy world. (Type 3)

There are free web-based systems that support collaborative drawing, so called "web whiteboards" (e.g. Ziteboard, LiveBoard, Explain Everything Whiteboard). Students may use their smartphones or tablets to create a collaborative visualization or algorithmic idea with this technology.

Concept cartoons

Concept cartoons (Naylor and Keogh, 1993) are comic-like illustrations depicting dialogs or thoughts of individual persons. The challenge is to explain a scientific issue using everyday language. Although they mainly consist of written text, concept cartoons are visualisations (type 4). They serve to express ideas on a topic from different points of view thus encouraging diversity of thinking. Concept cartoons in textbooks sometimes explicate typical misconceptions, that provoke discussion and falsification.

A concept cartoon-related task may be completely open. Example:

> "Tina visualizes a variable name by a sticky note that is put at an object. For example, the Python statement a = "car" she represents by the string literal "car" carrying a sticky note with the name "a" written on it. Draw a cartoon depicting Tina explaining this program."

A very common and less open version of concept cartoon-tasks provides drawings, empty speech bubbles and a start. The challenge is to continue the story. Figure X shows an example.

Finalized concept cartoons can later be used as as basis for discussions for instance on the limitations of metaphors that were used in the dialogs.

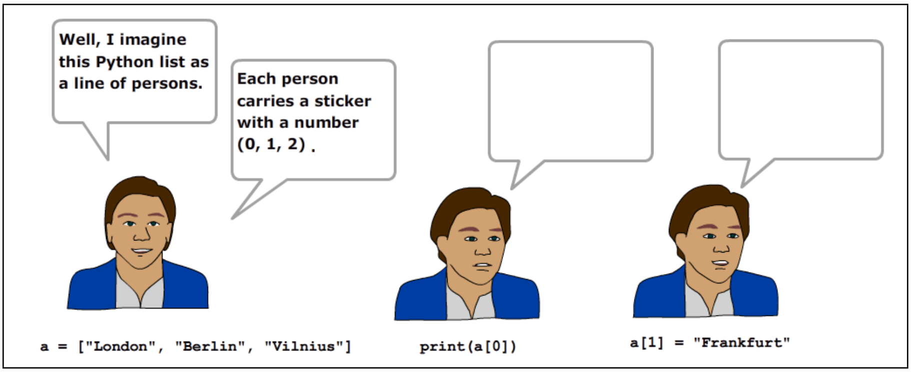

Well, I imagine this Python list as a line of persons.
Each person carries a sticker with a number (0, 1, 2).
a = ["London", "Berlin", "Vilnius"]
print(a[0])
a[1] = "Frankfurt"

Fig. 2. Concept cartoon.

Construction bricks

"Think with your hands" is the motto of representing abstract ideas with Lego. A participant picks a concept or question (like "How does a digital camera take a picture?") and creates (within a few minutes) a sculpture visualizing the idea (type 4) using all kinds of LEGO blocks. Then other participants have to find out what it is. Finally the creator explains his or her artifact. This activity should be done in small groups of four people sitting around a table. It includes a lot of communication and usually new and surprising ideas are exchanged.

There are two important rules for LEGO sculpturing that should be clarified first: 1) If you do not know what to build, just start building. The ideas will come later. 2) Everything, what you build is just fine.

Building with construction bricks requires a "warming up" to get in the mood of "thinking with your hands". A well-known procedure is the LEGO duck. Each participant gets six bricks and creates a duck in 60 seconds. There are many ways to solve the task. Even if you do nothing you have solved the task. In this case, the bricks are just the parts of a roasted and carved duck.

Instead of LEGO one can also use found material, things like pencil, ruler, paper sheet that are always present in a classroom. However, LEGO offers a greater diversity of material and the created artifact always looks nice and interesting.

Theatre play

People are used to role playing since childhood. Playing different roles is important for the development of social skills like empathy and tolerance of ambiguity. In CS education improvised role plays can be used to visualise algorithms or to give.

The many ways to organize role plays can be grouped in two types: Role plays that are prepared and performed independently by small groups and role plays in the style of "interactive theatre" that involve the whole audience and are controlled by the teacher or lecturer.

Role play in small groups could be a classic role play with dialogs and props, a mime or a "living statue". Here are examples for tasks of different types.

- **Type 1:** Create an algorithm for building a tower with LEGO bricks with four persons as quickly as possible.
- **Type 2:** Play a situation in which redundancy is important. Start with deciding where the story should take place (on the school yard, at a river, on the moon, …).
- **Type 3:** Play the execution of a function with two parameters. The parameters should represent things from this room. The function should return something.
- **Type 4:** Visualize the "Bubble Sort" algorithm in a silent theatre play.

An algorithm can be visualised or developed in the style of interactive theatre. Audience participation is quite established in theatre for children (Way, 1981). Some ideas of this tradition can be used in CS education. The lecturer encourages the audience to talk and play but still controls the development of the plot.

- **Type 1:** The lecturer develops an algorithm together with the audience. The audience tries out each idea immediately, discusses the solution and improves it, getting into an iterative process of constantly rethinking, adjusting and redoing.

Gerald Futschek (University of Vienna) is well known for this approach (see for example Futschek and Moschitz, 2010).

- **Type 4:** The lecturer directs a role play visualizing an algorithm that is difficult to understand, like a recursive version of the Quicksort sorting algorithm (Hoare). The props of the play are the algorithm, for example represented by a Python function definition, and a stack of big DUPLO bricks of different sizes and colors. Members of the audience play processors that can execute a function call according to the given code. The lecturer starts the play by passing the stack of bricks to one member of the audience. She or he scaffolds the play – and keeps the plot on track – by asking questions and initializing dialogues instead of explaining. Talking is essential for connecting the written code to the action of the play.

4.2. *How to Add Creativity to CS Tasks*

In this section we focus on potential of tasks from the Bebras challenge contest (Dagienė, 2017) as a good example of CS tasks which are not open ended. In these tasks, students plunge into a described situation which they must grasp, get to understand the concepts and terms that are used, find an informatics principle the task is based on and solve the problem using cognitive and thinking skills. These tasks are problem tasks, not creative. We can find several ways of involving creativity when solving Bebras task:

- Change the type of answer from multiple-choice to interactive.
- Move the Bebras tasks away from the computer.
- Take the Bebras tasks as inspiration for new tasks.

Change the Type of Answer of Bebras Task from Multiple-Choice to Interactive

Interactive Bebras tasks mean tasks where contestants give the answer by dragging objects on the screen or click on them. This allows to use tasks with more than one correct answer (and require to find at least one or all correct solutions).

Compare two versions of the same Bebras task about discovering Euler condition (title Even connections, 2017-PL-04):

A company has to connect 10 access points with data cables (lines) in one net according to a plan. Unfortunately this company has no technology to cut the cable to pieces so that it still worked. They can create one cable loop going through all access points and visit one point more than once.

Now it is impossible to solve this problem and it is necessary to remove some lines in the plan.

(multiple-choice version – Fig. 3 left): Which one of lines A, B, C, D is a part of the best solution of the problem if we want to remove as few lines as possible so that access points were connected without cutting the cable? Choose one of A, B, C, D. (Correct solution is C because it allows to remove only 3 lines in the graph; removing A, B or D lines requires to remove more than 3 lines in the graph).

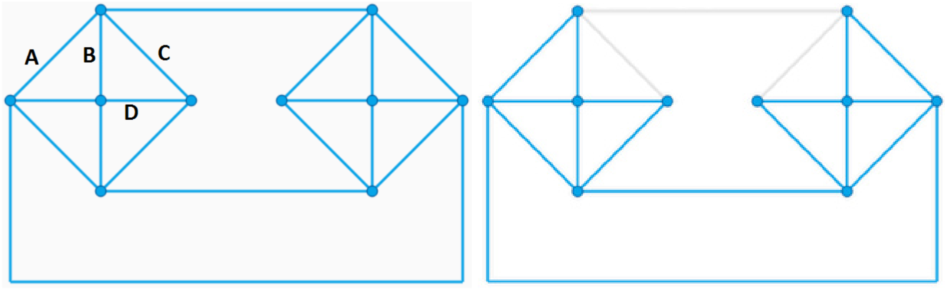


Fig. 3. Multiple-choice and interactive version of a Bebras task.

(interactive version – Fig. 3 right): Remove some of the lines by clicking on them so that the problem is solved by removing as few lines as possible. There is more than one correct solution.

Move Bebras Tasks Away From the Computer

Most of the Bebras tasks are ready to use without a computer. They have a story with objects as basics. We can bring these stories to life, and we can use these objects in our real world.

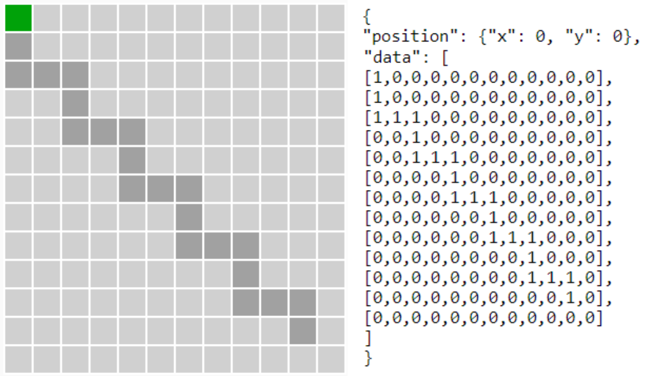
Instead of clicking, or dragging and dropping on the screen, we can move the objects, construct and “build” new buildings, sequences or objects. The success of problem solving strategies can be tested, and the teacher has the opportunity to follow along the strategies used, to help building and learning new strategies.

A typical example can be the task entitled Loading Lisa’s (2014-DE-08), where we need to put barrels onto the boats so that each boat gets loaded with as many kilograms of water as possible. On the screen, we can make an interactive drag and drop game. As an activity without computer, we might have more preparation ideas: using a magnet wand with magnet object; or we can prepare weight-objects (sacks with sands or cans with water with different weights), and two boats made of carton boxes. Students can move the weight objects, and experience with the solution – find the best strategy to solve the problem.

But not only drag-and-drop exercises can be converted.

- Students could become more creative when they can use theirs hands: building a construction from small parts – or the reverse version: find the small repeatable part in the big one.
- The real world can help solving 3D or other visualisation, such as origami, repeated buildings or activities with a paper or 3 dimensional objects.

In a challenge game or in an escape room, we can connect these activities: the first activity can give the instruction to the others, or more activity-results can add together the whole solution (or the path to the solution). Throughout these activities, participants can learn (or use) scheduling and sharing skills: they have the opportunity to work on tasks together. However, there are some problems that are best to be solved simultaneously, so participants need to define roles, and divide the activities into sub-activities.



```
{
"position": {"x": 0, "y": 0},
"data": [
[1,0,0,0,0,0,0,0,0,0,0,0],
[1,0,0,0,0,0,0,0,0,0,0,0],
[1,1,1,0,0,0,0,0,0,0,0,0],
[0,0,1,0,0,0,0,0,0,0,0,0],
[0,0,1,1,1,0,0,0,0,0,0,0],
[0,0,0,0,1,0,0,0,0,0,0,0],
[0,0,0,0,1,1,1,0,0,0,0,0],
[0,0,0,0,0,0,1,0,0,0,0,0],
[0,0,0,0,0,0,1,1,1,0,0,0],
[0,0,0,0,0,0,0,0,1,0,0,0],
[0,0,0,0,0,0,0,0,1,1,1,0],
[0,0,0,0,0,0,0,0,0,0,1,0],
[0,0,0,0,0,0,0,0,0,0,0,0]
]
}
```

Fig. 4. Interactive Bebras task – situation visible for contestants and initial data.

Taking Bebras Tasks as Inspiration for New Tasks

Bebras tasks could be a source of inspiration for changing something to create a new task. The new task is of course not creative but students are creative during its creation, they become "tasks creators" for other students, for competition participants.

- We can let students change a story or picture so that the problem would give a new look.
- Students can change part of the settings (e.g. initial word, number or string) of the task so that the kind of thinking will stay but the situation will change a bit.
- If an interactive task is prepared, the user can change the settings easily, the students can change the initial situation so that a new problem different from the original task occurs.

One example of such task is visible in Fig. 4: the interactive Bebras task Painting robot (2013-FR-05). The user has to write a program for the robot (green point in the corner) so that it goes through all dark fields in the playing field. The robot understands commands of cardinal directions movement and simple loops. Students can change initial data of this task (see Fig. 4 on the right) changing required shape to mark with possibility to create more complex or clever task than the initial one.

Do not expect that students as task creators will always be creative. Often they create something like variation of original task by making non-important changes in the settings, sometimes only by changing a story. But sometimes they could work with a completely new idea: more complex shape, nested commands, hidden connections between objects to be discovered...

4.3. *Adding CS to Creative Tasks*

It is well known that many user approach activities with ICT could be good starting points to inquiry, to discover computer science rules, to move form application to the core topic of computer science. E.g. painting in computer could move to thinking about pixels, colour depth, graphic formats. Writing texts could move to coding, data transfer, data compression. Work in vector graphics could result in logic operations and algorithms how to paint something. Creative work with digital technology can result in creative work in computer science, sometimes with unplugged activities.

We can develop creativity using problems with multiple solutions. Our experience show that it is hard for teachers to promote creativity in a quite new topic for students which unplugged computer science really is. One method is to show several ways of solving some open problems. Students, then, can imagine that such tasks allow different approaches of solving it, and might be more willing to create another.

Let us illustrate this approach on activity about how to code the data transferred. When electricity is supplied from external battery, Micro:bit can work unplugged without computer and can be used the user way when some program is downloaded to it. This activity uses Micro:bit as a tool for creating icons (picture).

Students use prepared program in which a cursor goes through all pixels in a matrix. They create their own icon by buttons A (light up), B (turn off) on the board (Fig. 5). Then students describe their icon by letters A, B, in the order they have touched these buttons, and give the code to the other student to create the icon in her/his device. In this part of activity, students are creative at user level. The role of Micro:bit is to check the rules (and of course to make the activity more attractive), this activity could be realized without this device.

In the second step, students create an icon using different given methods, as a path of movement of Karel the robot (a blinking point on the grid) controlled by two commands: forward (button A) and right turn (button B). Students could get that there are different methods of describing the icon using the codes of A’s and B’s, and might realize that without telling the method of coding it is hard to reproduce original work.

Finally students have to invent a “their own”, new secret code which can describe the icon so that only the receiver could reproduce it. Students can give different meanings to letters A and B (e.g. A sets painting pen up or down, and B moves the pen on the grid), or go through the grid in a different way (from the right, by columns ...). This creative activity could result in classroom discussion whether new methods of coding are correct, unambiguous and suitable for all icons. Students are creative as computer scientists in this part of the activity.

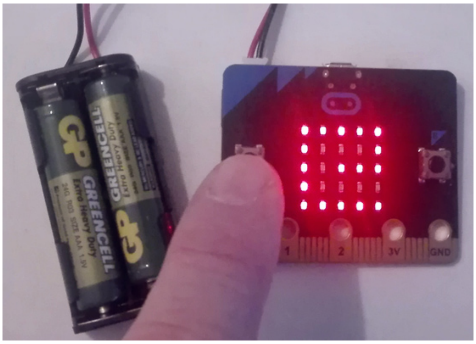


Fig. 5. Coding icons in Micro:bit.

4.4. *How to Embed Creative Unplugged Activities in the Learning Processes*

Creative artifacts are a contribution to the knowledge and culture of the learning community. Students should be aware of this value. In this section, we discuss a few ways of presenting the outcomes of creative activities, and how to embed them in the learning process of a class or lecture. On the web, artefacts can easily be published worldwide. However, according to the idea of the "local domain" of a learning community (see above) we focus on the presentation within the protected space of an educational institution.

Puzzle and Quiz

Examples of CS concepts, which students have created (type 2), may be used for a puzzle or quiz-like classroom activity. For example, after finding examples for list structures (Table 7) participants write Python list literals at the blackboard and for each literal the audience has to find out which real object in the room is modelled by it.

Similar puzzles could involve Lego artefacts or photos visualizing a concept (type 4), or stories that are related to an algorithm or concept (type 2).

A puzzle is an explicit cognitive challenge to the audience. Besides solving the puzzle, the audience is also encouraged to validate solutions, and to evaluate the quality of the visualization.

Exhibition

Artefacts that have some aesthetic value (drawings, sculptures, photos) can be collected and exhibited at a classroom wall or in a glass case. Example: One spot at the wall exhibits photos of all kinds of sorted sequences (jackets on hangers in a shop, floors of a building, books in the library, deck of cards). An exhibition – created by students – is not just a nice decoration but it reflects the students' everyday life, what is relevant to them and the way they think and feel.

The exhibits can later be used for programming exercises. For example, the "collection of sorted things" hanging at the wall might be a reason to think about how to model these entities using formal code. A deck of cards could be represented by a list of 52 tuples of the form (suit, card).

Performance

Roll play and mime performances are not easy to record. There are two major problems: It takes much time, competence and effort to record a performance and to create a good

Table 7
Finding examples for list structures

Python list literal (model)	Object in the classroom (reality)
`[[3, 4, 4], [3, 4, 2]]`	Two rows of tables with different numbers of students sitting at them
`[7, 7]`	Illumination at the ceiling consisting of two rows with seven lamps each

`[[0111, 1111, 1111],`
`[1011, 1111, 0011]]`
	Two rows of tables with four chairs each. The digits 1 and 0 indicate whether or not somebody is sitting on this chair

video. One has to think of camera perspectives, frames, sound and editing. You can find many videos documenting role plays on Youtube which are not only boring to watch but which are also difficult to understand.

The second problem is the personal rights of the actors. A video may be embarrassing for the actors, and it can only be published legally if the actors (and eventually their parents) have signed an agreement. So, in most cases, performances are live and remain undocumented. However, it is possible to take photos of scenes in way that the faces of the actors cannot be recognized.

If a live performance is dynamic (like an improvised role play), it may represent the idea of a new algorithm or a use case in an easily comprehensible way (type 1, 2 or 3). Such performance is done quickly, and it could be the starting point of a programming project. Example: Play sorting a deck of cards using straight selection and then write a sort function.

5. Conclusion

The spectrum of creative unplugged activities on CS can be grouped into four types: 1) Create an algorithm 2) Find an application 3) Find an example 4) Find a visualisation. This classification seems to be sufficient for most CS educators that have been asked in a questionnaire.

CS educators use creative unplugged activities for several reasons, especially as activating introduction into a new topic and to encourage students to think about CS concepts. Generally they see educational potential in all types. However, most educators rarely use these activities (a few times a year).

This may indicate that the repertoire of useful creative unplugged activities for CS education is still small and needs to be extended by the community of professional educators. Teachers developing new ideas for creative classroom activities might consider these points:

- There are different types of activities, which imply different ways of how to get creative. Being creative is joyful. Offering a diversity of tasks makes it easier for the individual student to find an activity to which she or he is able to contribute.
- There are many ways to express ideas (writing, drawing, building, playing, …). A student may pick a method she or he is good at to create.
- The result of a creative process is a new product (a new story, a new picture, a new sculpture, a new play etc.) that extends the "local domain" of relevant CS knowledge within a certain class at a certain school. This product can be presented and exhibited and is valuable for learning.
- There is a repertoire of successful creative activities in school education. Well known creative activities can be adopted to CS topics.
- There are many well thought – but not creative – unplugged activities in CS education including paper versions of Bebras tasks and exercises from the book "CS unplugged". Developers of educational material could take this wealth of ideas as inspiration and make closed and analytical tasks more creative.

References

Atmatzidou, S., Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems*, 75, 661–670.

Beecher, K. (2017). *Computational Thinking: A Beginner's Guide to Problem-Solving and Programming*. Swindon BCS.

Bell, T. (2018). CS Unplugged and Computational thinking, In: *Proceedings of Constructionism 2018, Vilnius, Lithuania*, p. 22 – 28.

Bell, T., Rosamond, F., Casey, N. (2012). Computer Science Unplugged and related projects in math and computer science popularization. In: Bodlaender H.L., Downey R., Fomin F.V., Marx D. (Eds.), *The Multivariate Algorithmic Revolution and Beyond*. Lecture Notes in Computer Science, vol 7370. Springer, Berlin, Heidelberg

Bell, T., Witten, I.H., Fellows, M. (2015). *CS Unplugged*. `www.csunplugged.org`

Boden, M. (2007). How creativity works, *Creativity East Midlands for the Creativity: Innovation and Industry conference*, 2007.

Rodriguez, B., Kennicutt, S., Rader, C., Camp, T. (2017). Assessing computational thinking in CS unplugged activities. In: *Proceeding SIGCSE '17 Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, p. 501–506.

Csikszentmihalyi, M. (1996). *Creativity: Flow and the Psychology of Discovery and Invention*, Harper Collins Publishers, New York

Curzon, P., McOwan, P.W., Plant, N., Meagher, L.R. (2014). Introducing teachers to computational thinking using unplugged storytelling. In: *ACM Proceedings of the 9th workshop in primary and secondary computing education*. 89–92.

Dagienė, V., Sentance, S., Stupurienė, G. (2017). Developing a two-dimensional categorization system for educational tasks in informatics. *Informatica*, 28(1), 23–44.

DeSchryver, M.D., Yadav, A. (2015). Creative and computational thinking in the context of new literacies: Working with teachers to scaffold complex technology-mediated approaches to teaching and learning. *Journal of Technology and Teacher Education*, 23(3), 411–431.

Feaster, Y., Segars, L., Wahba, S., O. Hallstrom, J. (2011). Teaching CS unplugged in the high school (with limited success). IN: *ITiCSE'11 – Proceedings of the 16th Annual Conference on Innovation and Technology in Computer Science*. 248–252.

Futschek, G, Moschitz, J. (2010). Developing algorithmic thinking by inventing and playing algorithms. In: Clayson, J.E., Kala , I. (Eds.) *Constructionist Approaches to Creative Learning, Thinking and Education: Lessons for the 21st Century, Proceedings Constructionism 2010*, Paris 16.-20. 8. 2010.

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

Haseski, H.I., Ilic, U., Tugtekin, U. (2018). Defining a new 21st century skill-computational thinking: Concepts and trends. *International Education Studies,* 11(4), 29.

ISTE, CSTA. (2011). *Computational Thinking in K–12 Education leadership toolkit*.

Korkmaz, Ö., Çakir, R., Özden, M.Y. (2017). A validity and reliability study of the Computational Thinking Scales (CTS). *Computers in Human Behavior*, 72, 558–569.

Lockwood, J., Mooney, A. (2017). Computational Thinking in Education: Where does it fit? A systematic literary review. arXiv preprint. *arXiv*:1703.07659.

Mishra, P., Yadav, A., Deep-Play Research Group. (2013). Rethinking technology & creativity in the 21st century. *TechTrends*, 57(3), 10–14.

Moreno-León, J., Román-González, M., Robles, G. (2018). On computational thinking as a universal skill: A review of the latest research on this ability. In: *Global Engineering Education Conference* (EDUCON) 2018, IEEE. 1684–1689.

Saunders, D., Thagard, P. (2005). Creativity in computer science. *Creativity Across Domains: Faces of the Muse*, 153–167.

Thies, R., Vahrenhold, J. (2013). On plugging "unplugged" into CS classes. In: *Special Interest Group on Computer Science Education*, Denver, 2013.

Voogt, J., Fisser, P., Good, J., Mishra, P., Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies*, 20(4), 715–728.

Weigend, M., Pluhar, Z., Juškevičienė, A., Vaníček, J., Ito, K., Pesek, I. (2018). Constructionism in the Classroom: Creative Learning Activities on Computational Thinking. *Constructionism 2018 Conference Proceedings*. URL: `http://www.constructionism2018.fsf.vu.lt/file/repository/Proceeding_2018_Constructionism.pdf`

Wing, J.M. (2017). Computational thinking's influence on research and education for all. *Italian Journal of Educational Technology*, 25(2), 7–14. DOI: 10.17471/2499-4324/922

M. Weigend studied Chemistry, Pedagogy and Computer Science at the University of Bochum and the University of Hagen and received a PhD in Computer Science from the University of Potsdam. He is a teacher at a secondary school in Witten, Germany, lectures at the University of Münster and he has taught Didactics of Computer Science at the University of Hagen for almost 20 years. He has published several books on computer programming and visual modeling.

J. Vaníček is associated professor and head of the Department of Informatics at the Faculty of Education, University of South Bohemia in České Budějovice, Czech Republic. He prepares primary and CS teachers and takes care of informatics teaching in primary and lower secondary schools and early age programming education. He works in expert group for innovation of national informatics curricula at the National Institute for Education.

Z. Pluhár is assistant lecturer of Faculty of Informatics at Eötvös Loránd University, Budapest, Hungary. She is member of the T@T (Technology Enhanced Learning) Lab and works mostly in teacher education. She is the head of Professional Community of Public Education at John von Neumann Computer Society. Her research fields are computational thinking, education of robotics and STEM. Since 2011 she has been organizing the Bebras informatics contest in Hungary.

I. Pesek is assistant professor of Computers in Education at the Faculty of Natural Sciences and Mathematics, University of Maribor. He is involved in teacher training of CS teachers. His main research interest are in computers science education and use of ICT in education.