Informatics in Education, 2020, Vol. 19, No. 4, 669–699

DOI: 10.15388/infedu.2020.29

Getting Inspired by Bebras Tasks. How Italian Teachers Elaborate on Computing Topics

Violetta LONATI
Dipartimento di Informatica, Università degli Studi di Milano, Italy
e-mail: lonati@di.unimi.it

Received: June 2020

Abstract. The Bebras challenge offers pupils and teachers an engaging opportunity to discover informatics, by solving small tasks that aim at promoting computational thinking. Explanations and comments that reveal the computing concepts underlying the tasks are published after the contest, and teachers are encouraged to use this material in their school practice. In this paper we present an exploratory study aimed at investigating how teachers can make use of Bebras material; in particular our interest is understanding whether teachers are able to identify, comprehend, and apply the computing concepts implied by Bebras tasks, and how they can integrate them into their teaching practice. We qualitatively analyzed teaching projects developed by Italian teachers during a workshop on computing education and based on Bebras tasks; the analysis shows that teachers are in general able to build upon the tasks soundly, but it also raises some critical issues.

Keywords: Bebras, teacher professional development, computational thinking, qualitative content analysis.

1. Introduction

Bebras is an international initiative[1] aimed at introducing pupils to informatics by means of a non-competitive contest. The challenge, that in 2019 engaged almost 3 million participants from 54 countries, consists of a set of *Bebras tasks*: fun small problems, which are based on computing topics but do not use technical language and do not assume any pre-knowledge in computer science (Dagienė and Futschek, 2008).

Even though the challenge is mostly run online, Bebras share most features of the CS unplugged approach (Bell and Vahrenhold, 2018): tasks can be solved also without the use of a computer (and indeed printed versions of tasks are used in some countries); learning by doing is the implied pedagogy; tasks do not focus on programming but cover the computing discipline in its broader meaning.

[1] The official description of the initiative is "International Challenge on Informatics and Computational Thinking", see `https://www.bebras.org`

At the end of the contest, all the proposed tasks are made available to students and teachers, together with the explanation how to solve them and some comments revealing the computing concepts underlying the tasks. In the intention of the Bebras organizers, this material serves mainly as a gateway into informatics as a scientific discipline, for both students and teachers. As a matter of fact, even though meaningful progresses have occurred recently, the presence of computing in schools is still mostly episodic; computing is a mandatory subject only in some countries, teachers and policy makers struggle at distinguishing between digital skills and the scientific computing discipline, and the need for training teachers is reported in many quarters (Barendsen *et al.*, 2015, McCartney and Tenenberg, 2014).

With their material, the Bebras community tries to compensate for this deficiency and to respond to teachers' different needs. Non-CS teachers are supported in the discovery of computing concepts that are presented with a nontechnical approach, whereas CS teachers are offered new engaging tangible examples and settings for typical CS problems. However, for a task to be useful as a teaching resource teachers need to be able to identify and correctly understand the computing aspect behind the task; moreover they need to sensibly elaborate on the topic, in order to integrate it in their school practice. Unfortunately, meeting either of these conditions can be hard since most teachers – especially in primary and lower secondary schools – do not have any formal education in computer science and lack the knowledge of the computing fundamentals.

Several studies have been conducted that involve Bebras tasks; most of them focus on the analysis of the computational thinking content of Bebras tasks, or use them as assessment items (see Section 2). However, there is a lack of studies that investigate how these tasks are perceived and used by teachers in their teaching practice. This kind of studies would provide insights on how teachers understand the computing concepts implied by Bebras tasks; hence, findings in this area would be beneficial for the evaluation of the effectiveness of Bebras tasks in teaching and learning, with implications to the more general discussion about the effectiveness of the CS unplugged pedagogy (Bell and Vahrenhold, 2018).

With this paper we contribute an exploratory study that aims at investigating this issue by analyzing teaching projects based on Bebras tasks; the projects were developed by Italian teachers during a professional development workshop. More precisely, our research questions are the following:

RQ1 Do teachers identify correctly the computing aspects behind Bebras tasks?

RQ2 Do teachers understand the computing aspects behind Bebras tasks?

RQ3 Are teachers able to sensibly develop the computing topics implied by Bebras tasks?

We analyzed the teaching projects using a qualitative descriptive method; an open-coding approach was used, where emerging categories were defined in an iterative process. The analysis shows that teachers are in general able to make good use of the tasks, but it also raises some critical issues. In particular we observed that the specific setting proposed in a Bebras task can negatively affect the comprehension of the underlying computing topic, especially when more than one computing concept is

mentioned simultaneously within the same task comment. We believe these findings give insights on the need of teachers, and this can contribute to both improving the quality of Bebras material and designing more effective professional development opportunities for teachers.

The paper is organized as follows. In section 2 we review the background and related work. In Section 3 we present the study design, describing in particular the context, the data collection process and the method used to analyze the data. We then present our results: in Section 4 we present the coding scheme that was defined during the qualitative analysis, and illustrate the emerged categories with examples taken from the data; the findings related to our research questions are presented in Section 5. In Section 6 we discuss our findings and relate them to previous work. Section 7 draws some conclusions and proposes ideas for future developments.

2. Background and Related Work

2.1. *Computing in Schools and Teachers Professional Development*

The role of computing in school curricula has recently received much attention in the education policies all over the world. In the US, a comprehensive standards for K-12 education has been proposed by the Computer Science Teachers Association (CSTA) in cooperation with the ACM (Seehorn, 2011). In the UK, the Royal Society Report "Shut Down or Restart" (The Royal Society, 2012) paved the way for the "Computing at School" curriculum (Computing at School, 2012), that establishes computing as a mandatory subject for all instruction levels starting from school year 2014–15. The "Informatics for All" proposal[2] has been presented by the associations Informatics Europe and ACM Europe to the European Commission with the aim of establishing informatics as an essential discipline for students in Europe at all levels. Similar efforts have been under way in several countries; a broad picture of the state of CS education worldwide can be found in (McCartney and Tenenberg, 2014).

In Italy computer science is not covered by curricular activities except for vocational schools (Bellettini *et al.*, 2014). A recent proposal by the Italian academic informatics community is meant to contribute to the development of informatics education in the primary and secondary school (Forlizzi *et al.*, 2018). Even though terms like *coding* or *computational thinking* are now commonly (mis-)used by Italian school policy makers, teachers have a very limited knowledge also of these terms, as testified by the study presented in (Corradini *et al.*, 2017) and conducted among primary teachers. Similar difficulties are encountered globally; as a paradigmatic example we mention the Royal Society Report "After the reboot: computing education in UK schools"(The Royal Society, 2017) which devotes two chapters to the importance of (and difficulty in recruiting) confident, well-qualified computing teachers, and their continued professional development.

[2] `http://www.informatics-europe.org/news/434-inf4all.html`

"Teacher professional development (PD)" has also become a frequent keyword (Menekse, 2015) for journal and conference papers in the area of computing education, where a recurring theme is the importance of integrating computing into the curriculum, especially in primary schools (Angeli *et al.*, 2016; Barr and Stephenson, 2011; Yadav *et al.*, 2014). However only a few studies investigate teachers' ability to understand and integrate such computing concepts in their teaching practice.

A survey was conducted among CS teachers in UK (Sentance and Csizmadia, 2017), in order to elicit their pedagogical perspectives; their answers to open-ended questions were qualitatively analyzed, and challenges and strategies were identified including, among others, unplugged activities, collaborative learning, conceptualization of tasks.

In (Dong *et al.*, 2019), teachers' products are analyzed in order to see how they infuse computing into their teaching. As in the current study, the products were prepared by teachers during a professional development, and included a description of some activities to be conducted in their classes; differently from the current study, the products focus only on programming (comprising a Snap! programming project and a document that explains the teaching intentions related to the project). Lesson plans developed by pre-service teachers are analyzed quantitatively and qualitatively in (Yang *et al.*, 2018; Sheridan *et al.*, 2020); in particular, in (Yang *et al.*, 2018) the aim is investigating how their computing and pedagogical knowledge develop over time. Findings reveal difficulties in conceptualizing and integrating computing concepts both from a disciplinary content and a pedagogical point of view.

2.2. *Bebras Tasks*

Every year a number of new Bebras tasks are created by a large group of experts, representing the many national organizations who belong to the Bebras community (Datzko, 2019). After the workshop, each local Bebras organization selects, translates, and adapts a selection of tasks from this pool, which are then used to run the challenge locally. In Italy pupils participate in teams, who attend the challenge via an on-line platform (Bellettini *et al.*, 2018a).

The tasks are equipped with explanations and comments; in particular, in the so-called "It's informatics!" section of tasks, the relevant computing concepts are named, and their relation to the task are made explicit; such material is made available to teachers and students after the contest. The task explanations can be defined as worked examples (Skudder and Luxton-Reilly, 2014), in that the task solution is built step by step and/or the reasoning used to reach the solution is developed gradually. Worked examples are proved to be effective instructional devices in learning science (Atkinson *et al.*, 2000); their use and effectiveness in CS education is discussed in (Skudder and Luxton-Reilly, 2014).

Bebras tasks have been subject of research in many studies: their conceptual content have been classified in (Barendsen *et al.*, 2015; Izu *et al.*, 2017), their quality and difficulty have been investigated in (Pohl and Hein, 2015; Lonati *et al.*, 2017; van der Vegt and Schrivers, 2019; van der Vegt, 2013); students' performance is the focus of a

multinational study (Dagienė *et al.*, 2014); proposals on how to integrate Bebras content into the curriculum have been described in (Dagienė and Sentance, 2016; Calcagni *et al.*, 2017); tasks have also been studied and used as assessment items (Araujo *et al.*, 2019; Solitro *et al.*, 2017; Hubwieser and Mühling, 2015; Hubwieser and Mühling, 2014; Chiazzese *et al.*, 2018).

Bebras tasks, especially those targeted at primary schools, usually present an informal playful setting where some characters has to solve some problem or challenge that is inspired by real informatics topics; in order to solve the task, pupils need to apply computational thinking skills without relying on previous computing knowledge. The underlying pedagogical approach is Dewey's *learning-by-doing* (Dewey, 1938) in that pupils learn computing concepts and acquire computational thinking skills by engaging with the tasks. These features are shared with the CS unplugged approach (Bell and Vahrenhold, 2018), and make the tasks suitable for adaptation to active learning units to be conducted in the classroom. That is, Bebras tasks can become the starting point for in-depth educational activities.

Despite the great popularity of the Bebras Challenge all over the world, little evidence is known about how teachers understand the computing content of Bebras material and how they use it in their school practice. In (Gujberova and Kalas, 2013), eleven future Slovakian informatics primary teachers (with good expertise in both computing and its didactics) are asked to analyze and comment on Bebras tasks, focusing on their difficulties and appropriateness for primary students, the cognitive operations implied by tasks, how they fit into the curriculum. Their answers have been qualitatively analyzed; the findings informed the design of tasks sequences that supports smooth learning process. Examples of learning activities derived from Bebras tasks are described in (Bellettini *et al.*, 2019; Dagienė *et al.*; 2019; Budinská and Mayerová, 2019). However, surveys conducted among teachers in Slovakia (Kalos and Tomcsanyiova, 2009) and Italy (Calcagni *et al.*, 2017) show that only a minority of them use the tasks beyond the challenge.

In (Dagienė *et al.*, 2016) two case studies are considered that show how teachers better understand certain informatics concepts by their involvement in creating Bebras tasks. Authors claim that the task creation process comprises constructionist and deconstructionist learning steps; a similar process occurs in our study, when teachers elaborate on Bebras tasks to create their teaching projects.

To the best of our knowledge no study investigates whether and how teachers personally elaborate on tasks in order to integrate them in their curriculum, which is indeed the intended goal of our exploratory study.

3. Study Design

In order to investigate how teachers elaborate on Bebras tasks, we analyzed teaching projects collected during a computing education workshop attended by in-service and prospective teachers. In this section we present the context of the study, the data collection process, and the method we used to analyze data.

3.1. *Context*

The workshop was held online in March 2020, it lasted two hours, and was part of a 48 hours course on computing education, spanning over 12 weeks. The workshop was held at the beginning of the course, more precisely during the second week.

The course is offered annually in the CS program of a public University in Northern Italy, is co-taught by three instructors, including the author of this paper, and is designed by a socio-constructivist approach (Glasersfeld, 2001). The content and pedagogical approach of the course are presented in (Bellettini *et al.*, 2018b).

The course is optional and enrolls students with a computer science background (most of them are CS graduated students) who are interested in a teaching career, even if they have not openly chosen it yet. The course is also open, as a PD opportunity, to in-service teachers from all school levels, who may not have any formal training in informatics.

The workshop was administered via a video-conference tool, and virtual break-out rooms were used to allow group work under the supervision of the instructors. In-service teachers and CS students worked in separate groups both during the workshop on Bebras and during the first week of the course.

Prior to the workshop, video materials were published to be seen beforehand. In the video recordings, lasting about 90 minutes overall, the course instructors presented the Bebras challenge and the use of Bebras tasks as a teaching resource. In particular they covered:

- A general introduction to the challenge.
- Some examples of Bebras tasks with a discussion of the implied computational thinking aspects.
- The detailed presentation of three examples of teaching activities that teachers designed and proposed in their classes, inspired by Bebras tasks.

During the workshop, attendants were asked to solve in pairs some new Bebras tasks, to write the explanation how to solve the task in a language easy to understand by pupils of the appropriate age, to peer-evaluate the explanation written by other groups, and to discuss the computing related aspects of the tasks they worked on. During their work, instructors were available for feedback.

3.2. *Data Collection Process*

The teaching projects analyzed in this paper were prepared as homework at the end of the workshop. The assignment was not mandatory and did not give any credit or mark, but attendants were told they would receive feedback from instructors concerning their projects. The submission was due a couple of weeks after the workshop.

Attendants were asked to choose a Bebras task they had not worked on yet, and to "*develop the implied computing aspect from a teaching perspective*". The assignment was quite open and in particular it did not establish further specifications on either the content, or the setting or duration of the teaching activity, or the format or length for the

project presentation. As examples, teachers were suggested "*to elaborate on the computing aspect to make it more accessible to pupils, or to prepare a lesson which takes inspiration from the task, or to design a new learning activity starting from the task*".

Primary and lower secondary non-CS teachers, having no or a little CS background, could choose among a selection of published tasks, hence they had access to the explanations and comments about the computing aspects (the "It's informatics" section of the task). On the other hand, both CS students and secondary schools CS teachers could choose among a selection of unpublished tasks and were given only the task texts, so they first were required to figure out the correct answer and to identify a computing aspect relevant for the chosen task.

Teachers and students were encouraged to work in pairs whenever possible, and they could freely choose their group mate.

The projects were to be submitted in digital forms. All attendants chose to prepare either text documents or slide presentations. We collected 14 projects prepared by 24 attendants: 6 projects by 9 non-CS teachers, 2 by 3 CS teachers and 6 by 12 students (one of these students was actually part-time employed as a CS teacher as well). The chosen tasks with the related computing aspects are summarized in Table 3.2 and fully reported in the Appendix.

In what follows, if we don't need to distinguish between in-service teachers and students, we will simply use the term "authors".

3.3. *Analysis Method*

We analyzed the collected projects by using *qualitative content analysis* (Mayring, 2014), a systematic approach for exploring, coding and categorizing textual information.

The purpose of content analysis is to describe the features of the document's content by examining what is written and how. An inductive open coding approach was used, where categories related to the research questions emerged in an iterative process.

In the first phase only three broad categories were considered, which correspond to the three research questions:

1. A computing topic is explicitly mentioned and/or addressed in the project.
2. There is some evidence that the computing concept has been misunderstood.
3. Supplementary material is proposed that is new with respect to the original task material.

During this phase interesting elements were observed. In particular, the following specialized codes were defined and used to classify the supplementary material.

- **Activity:** supplementary activity/task to be carried out by pupils.
- **Application:** the computing concept is applied in a different context.
- **Explanation:** a supplementary explanation is provided for the solution of the Bebras task.
- **Theory:** a supplementary (general, theoretical) presentation of the computing topic is provided.

Table 1
List of the chosen Bebras tasks, with age category, the underlying computing aspects, number of collected projects and background of projects' authors

Bebras ID	Task title	Age	Computing topics	n.	Authors
2019-LT-07	Snowmen	8-10	heap, stack	2	non-CS
2019-CH-13d	Stamps	8-10	complexity theory	1	non-CS
2019-SI-03	Space travel	10-13	graph, finite state automaton	1	non-CS
2019-CH-11c	Cloud communication	10-13	encoding, redundancy	1	non-CS
2019-LT-09	Lockers	10-13	binary code	1	non-CS
2019-BE-07	Greener flight route	13-18	graph, minimum spanning tree	1	CS
2019-KR-04	Buying shoes	13-18	binary search	2	CS
2019-CA-04	Delivery service	13-18	graph, Hamiltonian path	1	CS
2019-BE-04	Friendship bracelets	13-18	formal grammars	4	CS

In a second phase, recurrent features among projects were used to refine the above initial categories and to define the final coding scheme. The coding scheme, which is described in Section 4, was then applied to all teaching projects, and some quantitative indicators were computed. We will discuss the related findings in Section 5.

4. The Coding Scheme

The coding scheme that emerged during the analysis is summarized in table (see Fig. 1). The scheme comprises twelve categories; for each category the table shows the relevant research questions and lists the potential codes, together with a brief description. The categories can be roughly grouped into two broad themes:

- **Computing:** categories concerning the computing aspects addressed in the projects
- **Pedagogy:** categories revealing the pedagogical approaches that informed the projects' design.

To illustrate such categories and codes, we will now present some examples taken from the data. We recall that the Bebras tasks inspiring the teaching projects are all available in the Appendix.

4.1. *Computing Categories*

One of the categories under this group is "explicitness", that considers whether computing topics are explicitly mentioned in the teaching material or are implicitly addressed without explicit mention. For instance, in one of the projects based on task "Snowmen", a pair of non-CS primary teachers proposed the following activities: 1) using cards, simulate the process that assigns hats to snowmen according to their

Gategory group	Category	RQ			Potential codes	Description
		1	2	3		
Computing	expliciteness	x			explicit	the computing aspect is explicitly named or addressed
					implicit	the computing aspect is implied by the supplementary material
	consistency	x			suggested	the computing aspect is mentioned in the "its' informatics" section of the original Bebras task
					other	other computing aspects are mention that are not included in the "it's informatics" section of the task
	use of CS terminology		x		proper/improper terminology	specific CS terminology is correctly/incorrectly used to refer to the addressed computing aspects
	quality of supplementary material		x	x	correct/incorrect	the supplementary material is correct/incorrect
					precise/vague	the supplementary material is precise/vague
					complete/incomplete	the supplementary material is complete/incomplete
					clear/difficult to understand	the supplementary material is clear/difficult to understand
	connection		x	x	soundly/badly	the supplementary material soundly/badly relates to the addressed computing aspect
Pedagogy	type of supplementary material			x	activity	supplementary activity/task to be carried on by pupils
					application	the computing aspect is applied to a different context than the task
					solution	an explanation of the task solution is provided
					topic	the computing topic is presented
	build upon			x		the supplementary activity is built upon the original Bebras task
	teaching strategies			x	active learning	discussion, group work, manipulation, simulation
					traditional	lecture, exercise
	proximity			x	within/out of the reach	the supplementary material is/is not within the reach for the pupils
	other subjects			x		other school subjects are mentioned
	learning outcome		x	x	related/unrelated	learning outcomes are mentioned that relate/do not relate to the computing aspect
	goal			x	problem solving	the goal of the supplementary material is promoting the problem solving skills needed to solve the task
					concept	the goal of the supplementary material is learning a computing concept

Fig. 1. The final coding scheme used to qualitatively analyze the teachers' project. For each category the relevant research questions are selected, and the potential codes are listed together with a brief description. The categories are grouped into two broad themes.

orders in the stack of hats and in the queue of snowmen, and 2) build situations where the algorithm gets stuck and cannot assign all hats properly (because they are in the wrong order). These are clearly activities with meaningful computing content – the *execution and debugging* of an algorithm – even though teachers neither identify nor name it explicitly as a CS topic.

Category "connection" refers to the fact that the supplementary material is soundly or badly related to the mentioned computing concept. For instance, in the project based on tasks "Stamps" one reads: "The idea is using the task to deepen the procedure of arithmetic expressions, suggesting pupils to use procedures that reduce the number of operations to be performed". This sentence was coded as badly connected to the original Bebras task.

Category "quality of supplementary material" in the "computing" group comprises potential codes such as correct/incorrect, precise/vague, complete/incomplete, clear/difficult to understand. Fig. 2 shows an example of content found in a project based on task "Delivery service". The content was coded as incorrect since the title is "Dijkstra algorithm" but the diagram does not illustrate the algorithm; instead it shows the tree of all paths (together with their costs) of the graph, starting from node A.

Another example of content that was coded as incorrect was found in the project based on task "Space travel", where a finite state automaton was used to model the relations between the components of a system (mistaking "states" for "components" and "transitions" for "passages"), see Fig. 3.

An example of content that was coded as improper use of terminology is the reduction of technical terms "stack" and "queue" to synonyms for vertical and horizontal arrangements of elements, as in the following segment, found in one of the projects

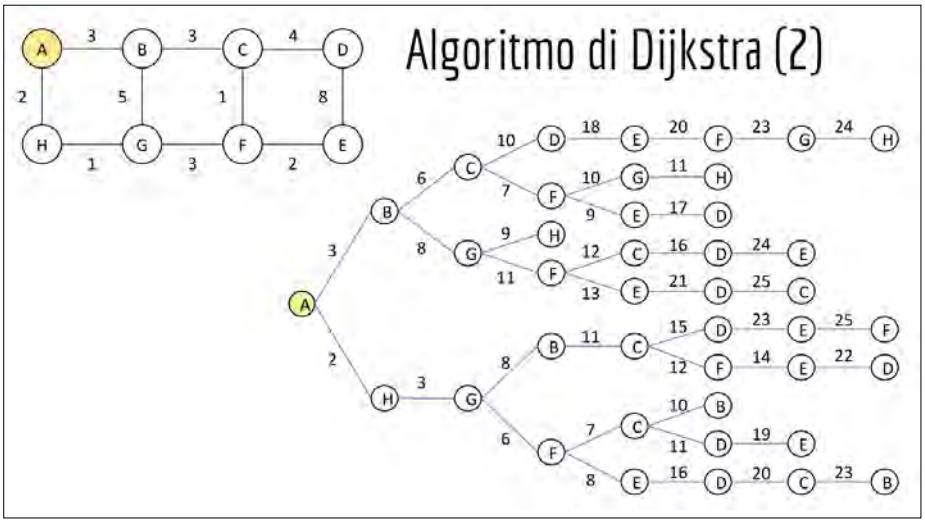


Fig. 2. Example of content that was coded as incorrect, since the title is "Dijkstra algorithm" but the diagram shows the tree of all paths (together with their nodes) of the graph, starting from node A.

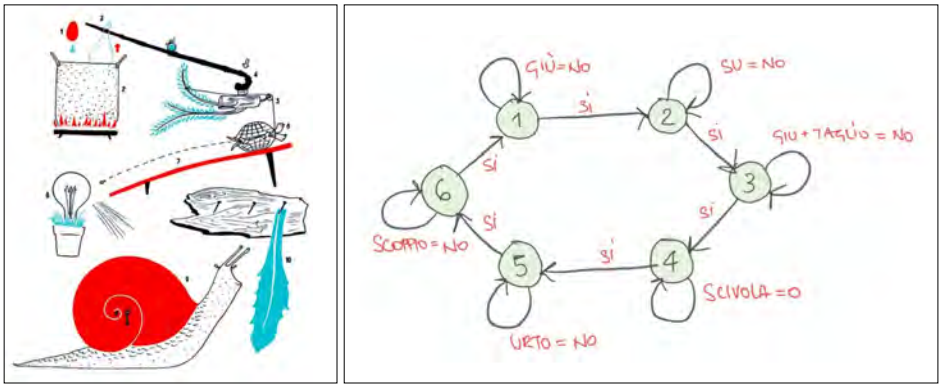


Fig. 3. One of Munari's useless machines (on the left, excerpt from *Bruno Munari, Munari's Machine, Corraini Edizioni*) is modeled with the graph on the right; the content was coded as incorrect since transitions are used to model the relations between the components of the system and not the transitions between states of the system.

based on task "Snowmen": "we will introduce the concept of *stack*, hence vertical arrangement of elements, and the concept of *queue*, hence horizontal arrangement of elements".

Category "explicitness" and "consistency" provide elements to answer RQ1, while content classified by codes as incorrect, improper terminology, or badly connected is considered evidence of bad understanding of some computing concepts (RQ2).

4.2. *Pedagogy Categories*

Beyond the distinction among different kinds of supplementary material, we have codes that reveal the pedagogical approach that informed the project design. Category "teaching strategy" includes two codes: traditional (as a lecture or an exercise), or active learning (e.g, group work, manipulation of small objects, discussions). For example we coded as active learning (manipulation) the proposal to use a cardboard strip in task "Cloud communication", in order to "hide one or more rows at a time (in the table with cloud codes) and check if the remaining combinations still allow to distinguish the messages unambiguously" (see Fig. 4).

Category "proximity" refers to Vygotsky's zone of proximal development (Vygotsky, 1978) and describes the fact that the supplementary activities are suitably scaffolded and within the reach for pupils. An example of content that was coded as out of reach was found in the project based on task "Space travel": pupils are asked to design a finite state automaton for a complex situation, while in the original Bebras task a finite state automaton was given, and simulating it was enough to solve the task.

Category "goal" refers to the taxonomy that distinguishes between knowledge, skill, and competency; category "learning outcome" refers to the fact that relevant/irrelevant expected learning outcomes are stated and/or discussed in the project.

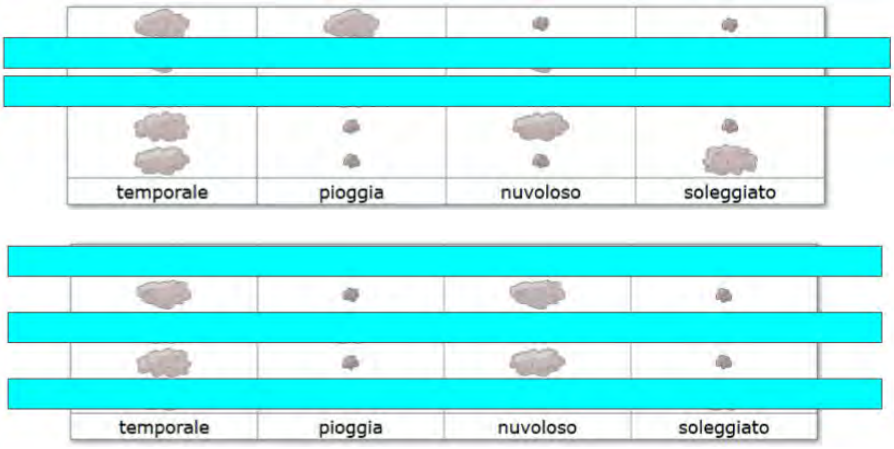


Fig. 4. The use of a cardboard strip is proposed to hide one or more rows at a time in the table of task "Cloud communication".

5. Findings

The description of findings is organized according to the related research questions.

5.1. *Findings for RQ1: Identify Computing Topics*

All non CS-teachers addressed the computing aspect that was mentioned in the "It's informatics section" of the chosen Bebras task. Three of them addressed in their project also other computing topics or skills, mostly implicitly. Among them we find the activities on simulating and debugging algorithms, already described in Section 4. Other computing aspects covered by non-CS teachers' projects and not already mentioned in the "It's informatics section" are: increasing/decreasing order and sorting (task "Snowmen"); using a table to represent data (task "Snowmen"); combinatorics, byte and measures for data storage (task "Lockers").

On the contrary, CS students/teachers did not have access to the "It's informatics" section of their tasks and needed to uncover the implied computing aspect. Five over eight groups identified the topic that was indeed mentioned in the "It's informatics section", and another one identified it partially (the group correctly identified and named the graph data structure in "Delivery service" but didn't recognize that the task presented an instance of the Hamiltonian path problem). For task "Friendship bracelets", two groups did not recognize that the task was about formal grammars; instead they associated the task with recursion and linked lists. The first association is correctly supported, whereas the connection to linked lists is arguable; in particular the group wrongly described as a list also the right-hand-side of a grammar production containing an OR clause, which is not appropriate.

Moreover, five among these CS groups also mentioned other applicable computing aspects: importance of sorting for binary search, counting the number of steps and logarithms, computational complexity (task "Beaverella"); parsing (task "Friendship bracelets"); optimization problems (task "Greener flight routes").

Finally, one CS students group mentioned greedy algorithms and in particular Dijkstra's algorithm with respect to task "Delivery service". In this case the connection to the task is weak, since the Hamiltonian path problem proposed in the task cannot be solved by either approaches.

5.2. *Findings for RQ2: Understand Computing Topics*

The use of specific CS terminology happens to be correct and accurate in most (6 over 8) CS-authors' projects, and imprecise or incorrect in most (4 over 6) non-CS teachers' projects.

In CS student/teachers' projects we observed errors in advanced topics: confusion between "recognition" (with automata) and "generation" (with grammars) of formal languages, in task "Friendship bracelets"; improper or vague expressions related to optimization problems ("generalize the function that optimizes the graph", "search for a configuration of the graph that minimizes a certain property"), both in task "Greener flight routes".

In projects written by non-CS teachers we observed specific computing terms used with the common-sense meaning that the same terms have in everyday language. For instance, in both projects based on task "Snowmen", the concepts of *queue* and *stack* appear to be misunderstood, since these data structures are superficially associated with the concept of vertical (stack) and horizontal (queue) lines. This misconception fits perfectly the common-sense meaning of Italian words "pila" (i.e., "stack") and "fila" (i.e., "queue") used in the text of the task.

Another similar error in the use of specific CS lexicon concerns the meaning of expressions "automaton" and "transition" in task "Space travel" designed around the idea of finite state automata. Also in this case the common-sense meaning of the terms mislead the teacher, who wrote definitions that are not correct in this context: "with term *automaton* we refer to a system that can perform an activity without human intervention" and "transition are passages among the component of the systems", instead of passages among states of the system.

We now consider the quality of the supplementary material proposed in the projects. All applications proposed by groups of CS teachers/students were correct, whereas in two cases the applications by non-CS teachers were critical: complexity theory was referred to counting the number of parenthesis in an arithmetic expression; a finite state automaton was wrongly used to model the relation between components of a system (see Fig. 3).

Half of the projects that presented explanations or general presentations had a very high quality as far as the correctness, completeness, and clarity were concerned. In the remaining projects we observed some issues; those are the same projects where we ob-

served an incorrect or imprecise use of CS-specific language as well. The severest issue concerning such explanations and presentations was related to the Dijkstra algorithm, which finds the shortest path between nodes in a graph. The algorithm was improperly mentioned in relation with the Hamiltonian path problem of task "Delivery service" (see also Fig. 2).

In all projects, authors explicitly related supplementary materials to the addressed computing topic. In most cases, this connection is sound and well supported. Notable exceptions are the already mentioned explanation of task "Delivery service" and the activities inspired by task "Snowmen" and "Space travel", where there is evidence that the computing concepts were not fully understood, as supported by the above findings about the use of CS terminology.

5.3. *Findings for RQ3: Develop Bebras Computing Topics*

We now consider the type of supplementary material proposed in the projects, as described in Section 3.3: application, explanation, theory, and activity.

- Half of the projects presented some application of computing concepts to different contexts than those presented in the tasks. The same ratio occurs in projects prepared both by non CS teachers (3 over 6) or and CS teachers/students (4 over 8).
- The explanation on how to solve the task was never included in projects written by non-CS teachers, whereas 5 over 8 CS teachers/students proposed such an explanation.
- An explicit presentation of a computing concept occurs in 8 projects. Again, the difference is significant between CS and non CS teachers. More precisely, only 1 over 6 non CS teachers' projects included a presentation of the computing topic (mostly rephrasing the "It's informatics" session of the original Bebras tasks) whereas 7 over 8 CS students/teachers' groups did include such a presentation.
- All projects except two proposed that pupils carry out one or more new tasks; in particular this is true for all project presented by teachers, since the only two projects that do not comprise this kind of supplementary material were prepared by groups formed only by students.

To investigate the pedagogical approaches used by teachers we consider which kind of activity the teachers proposed and how they sequenced them, how and when the Bebras task is used in this sequence, what connections are made with other school subjects, what learning outcomes are considered.

Many of the tasks proposed fall into the wide category of active learning tools and strategies: group work, discussions, simulations, problem solving open tasks, role play, manipulation of small objects. Five versus one non-CS groups favored the active learning approach, whereas among CS groups the trend is reversed, since five over eight used traditional teaching methods such as transmissive explanation, possibly followed by exercises.

In two cases only, the supplementary material was developed around a computing idea with no reference to the setting of the original Bebras tasks. All other projects where built upon the original task, which was included in different ways:

- As the initial activity for a learning unit, i.e., as the starting point to introduce the computing aspect (4 projects).
- As a final task to be solved after preliminary activities and/or explanation that present the computing aspect (1 project).
- As the central or recurring example/setting for a sequence of different activities that work on and strengthen the same computing concept or skill (7 projects).

Only five projects contained some reference to other school subjects (4 maths, 1 technology); four of them were written by non-CS teachers. Only a minority of projects explicitly stated their goals or learning outcomes, but they could be inferred (except in one case, where the description was too vague). We found both knowledge goals (knowing the computing concept implied by the task) and skill goals (mainly, developing the problem solving skills needed to solve the tasks) for their pupils. All CS authors except one claimed knowledge goals, but half of them actually aimed at both kinds of goals, with only one project focusing only on problem solving. Among non-CS teachers the knowledge goal appears to be less important, since 4 over 6 of them focused only on operational and problem solving skills.

In most cases, the activities proposed in the projects are within the reach for pupils, i.e., in their zone of proximal development (Vygotsky, 1978), and are suitably scaffolded. There are some projects though where this is arguable; for instance in one case pupils are asked to design a finite state automaton for a complex situation, while in the original Bebras task a finite state automaton was given and it was enough to simulate it in order to solve the task. In another case, pupils are asked to design a formal grammar for a given language, while the original task and the supplementary material all focused on recognizing if a string is generated by the grammar.

6. Discussion

The findings presented in the previous section are summarized in Fig. 5. Before discussing them, we acknowledge some limitations of the study.

6.1. *Limitations*

First and foremost, the sample size was rather small. Moreover, some of the subjects involved in the study were not in-service teachers but prospective teachers or, more precisely, CS students interested in a career in teaching; this lack of experience can have an impact in their ability to design appropriate learning activities.

The experience with Bebras varied greatly among the attendants of the workshop, with teachers who had participated with their classes several times and others who did

Findings:

RQ1 - Do teachers identify correctly the computing aspects behind Bebras tasks?

1. CS teachers/students groups were in general able to associate the Bebras task to an appropriate computing aspect.
2. Non-CS teachers were often able to associate the Bebras task with other relevant computing aspects, beyond those mentioned in the "It's informatics" section.

RQ2 - Do teachers understand the computing aspects behind Bebras tasks?

3. CS terms that also belong to everyday language may mislead non-CS teachers and prevent the correct understanding of computing concepts.
4. The comprehension of a computing topic by non-CS teachers may be hindered when they rely only on the single specific example proposed in a Bebras task.
5. Non-CS teachers may build confused knowledge of different computing concepts if they are presented simultaneously in association with the same task, that is, with a single setting or example.

RQ3 - Are teachers able to sensibly develop the computing topics implied by Bebras tasks?

6. Teachers use a Bebras task mostly as a recurring example/setting for a sequence of different activities.
7. The pedagogical approach informing the projects differs between CS and non-CS authors:
   - non-CS teachers choose more often an active learning approach;
   - non-CS teachers aim more often at their pupils to develop skills instead of acquiring knowledge.
8. There is a correlation between the comprehension of the computing concepts and the capability of designing activities that are within the reach for pupils.

Fig. 5. Summary of findings.

not know the Bebras challenge prior to the workshop. However, this is not critical in this study since Bebras tasks aim at being immediate to understand also for people with no computing background.

Finally, the assignment was quite open and participants were free to choose how to structure their project and which features to include; for instance i) some of them explicitly mentioned, summarized, explained or presented the computing aspect involved by the task, whereas others left it implicit, and ii) some of them explicitly discussed how the task could be solved and/or reworked the explanation given with the task, whereas others did not, even when they proposed activities that are clearly aimed at supporting students through the problem solving process required by the task. Due to this, in some cases only partial information could be extracted from the projects, with respect to our research questions.

6.2. *Identifying, Understanding and Developing Computing Concepts*

Identifying. In their projects participants generally addressed computing aspects that were relevant with respect to the chosen task. This was the case of all non-CS teachers (who were given the full explanations and comments for the tasks) and most CS students/teachers (who were given only the task text). In a few cases some non-relevant topics were mentioned, mostly regarding advanced CS topics as graph algorithms or data structures, which raises some doubts about the solidity of teachers' CS background and confidence.

On the contrary, in non-CS teachers' projects we found elements that could be associated with computing aspects, beyond those present in the original task. Even though the teachers do not seem to be aware of that connection, this is promising and confirms the possibility of integrating computing concepts with teachers' prior knowledge and skills.

Understanding. Whether the teachers properly understood the computing concepts implied by the tasks is more debatable. In some cases we found evidence that some topics were misunderstood or wrongly generalized. This happened for instance when the technical definition of CS terms clashed with the common-sense meaning of the same terms. This is the case of both projects based on task "Snowmen", inspired by stacks and queues. Despite the fact that the "It's informatics section" of the task contained a definition of these data structures and of the LIFO and FIFO paradigms that distinguish one data structure from the other, these terms were used by the teachers simply as synonyms of vertical and horizontal arrangements. It's worth noticing that, in the problem solving process required by the task, this distinction between LIFO and FIFO was not as important as matching the horizontal ordering with the vertical one, which is indeed the concept the teachers focused on in their teaching projects.

Another interesting mistake was found in the project based on task "Space travel", where the teacher tried to apply the notion of finite state automaton to a new different setting. The resulting application was incorrect, though, as not encompassing the idea of transitions of a system from a state to another. In this case the "It's informatics" section of the Bebras task presented two main concepts, namely graphs and finite state automata; our interpretation of the error is that the teacher collapsed these two concepts in one, as if every graph represented a finite state automaton, which of course is not true in general.

In all the above cases, the errors are somehow compatible with the knowledge provided by the Bebras material, and we interpret it as the result of a wrong generalization from the specific setting or example given in the task. This is indeed one of the warnings when applying either the learning-by-doing or learning-by-example pedagogies; *learning transfer* from specific examples or settings can happen only if those examples are varied, and discussed with respect both to each other and the recurrent schemata they imply (Atkinson *et al.*, 2000, Skudder and Luxton-Reilly, 2014). Clearly this is hard in the context of Bebras tasks, which need to be kept short and single by nature.

Developing. As for the supplementary material developed by the workshop attendants, we found differences between CS and non-CS teachers. The explanation on how to solve the task was never included in projects written by non-CS teachers, whereas half of CS teachers/students proposed such an explanation. We may interpret this difference by recalling that only the former had access to the published explanation of the solution. Thus, non-CS teachers might have considered the provided explanation clear enough, whereas the CS students/teachers needed to develop their own solutions, and most of them decided to include an explanation for it in their project. However, some of them did not include any explanation; a possible interpretation is that they did not deem it necessary given their prior knowledge of the implied topic; this interpretation is supported by the fact that all projects by these authors show evidence of a good understanding of the addressed computing topic.

An explicit presentation of a computing concept occurs much more likely for CS teachers/students than in non-CS teachers. Our interpretation for this difference is that CS teachers and students may have a higher confidence in their computing knowledge than non-CS teachers. Moreover, CS students' groups often proposed slide presentations with general (theoretical) presentation of the computing topic implied by the task, whereas teachers never used such a lecture form. We attribute this to the different pedagogical background and expertise among teachers and students, the latter being indeed very used to this kind of lecture in university classes.

Considering the pedagogical approach, we noticed that CS teachers used more often a traditional approach; on the contrary, non-CS primary teachers all showed an approach based on active learning, and aimed at their pupils to develop skills more than acquiring knowledge. This is consistent with the expected learning outcomes and pedagogy recommended in Italy for primary schools.

A final observation concerns the teachers' capability of designing new activities that are within the reach for pupils. In general we noticed that this capability is stronger when the comprehension of computing concepts appears solid. In particular, there are two cases where pupils are asked to solve "inverse" problems that distance from what was done previously, in that they require very different skills to be solved even if the computing topic is the same. For instance in one case pupils are asked to model a complex situation with a finite state automaton, while in the original Bebras task a finite state automaton was given and simulating it was enough to solve the task. In another case, pupils are asked to design a formal grammar for a given language, while the original task and the supplementary material all focused on recognizing if a string is generated by a given grammar. In both these cases, we had also found evidence that the computing topic was not well understood by the projects' authors (incorrect application of the concept in one case, and improper use of terminology in the second case). We interpret this in the framework of PCK (Pedagogical Content Knowledge) theory (Shulman, 1986), in that the capability of teachers of proposing sensible tasks and trajectories for their pupils strongly rests on both pedagogical knowledge/skills and knowledge of the taught discipline.

7. Conclusion and Future Work

In this study we carried out an exploratory investigation on how teachers can profit from Bebras material. We collected teaching projects based on Bebras tasks and qualitatively analyzed them. Our findings, in connection to our research questions, are summarized in Fig. 5.

In general, we found that teachers were able to identify and understand the computing concept underlying the tasks, and to develop sound learning activities for pupils, which confirms the potentiality of Bebras material as a teaching resource, as claimed in (Dagienė and Sentance, 2016, Calcagni *et al.*, 2017). Nevertheless we identified some potential issues, in particular for those teachers who do not have a background in computer science. Bebras tasks do nicely introduce teachers to new computing concepts, however the presentation of such concepts rests mostly on one single example (precisely the one in the task), which may make it difficult for teachers to correctly generalize the concepts, or even mislead them due to the specific features occurring in the task settings. This effect is further amplified when a single example or setting is used to introduce more than one computing aspect. Moreover, the CS concept is better understood when it is really required to solve the task and is not just superficially associated with the task setting. These findings confirm that the knowledge derived from the research on worked examples (Atkinson *et al.*, 2000) applies to this setting as well, and suggests that the instructional principles developed therein should be used as references also when creating Bebras tasks.

Thus, despite the small sample of the study, some recommendations can be drawn for Bebras' authors that can be easily generalized to other CS instructional material and design of teacher PD development –especially those based on unplugged activities:

- To make sure that learners will run into the intended computing topic when solving the task.
- To avoid overloading a task with different related computing meanings, that is, to choose one and only one computing concept to associate with the task.
- To include also other meaningful examples or settings when presenting a computing topic related to a task, and to highlight similarities and recurrent patterns among examples.

As another contribution of this paper, we believe that the methods used here could be applied to analyze other instructional material designed by teachers – not necessarily based on Bebras material– thus helping increase the knowledge on whether and how teachers can understand and integrate computing in their teaching practice. As a future work, we plan to extend the study by involving a greater number of teachers and providing them beforehand with a template for presenting their teaching ideas, in order to get more complete and structured projects.

As a final note, we remark that the results of this study resonates with a wider issue concerning the professional development of teachers, which afflicts in particular our discipline. In fact, and differently from most other disciplines, teachers usually have not been exposed to computer science during their own education. Hence, attend-

ing professional development courses might not be enough to build a solid computing knowledge. For this reason we claim that, until a new generation of teachers with a different CS background will take the floor, when designing teachers' PD opportunities and instructional material we need to carefully combine two fundamental components: on the one hand the methods and strategies to teach informatics and computational thinking, and on the other hand a constant reflection of what computer science is and its fundamental principles.

Acknowledgements

The author would like to thank the International Bebras community and the fellow Italian Bebras organizers for the great effort spent in producing such an interesting material.

References

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., Zagami, J. (2016). A k-6 computational thinking curriculum framework: Implications for teacher knowledge. *Journal of Educational Technology & Society*, 19(3), 47–57.

Araujo, A. L. S. O., Andrade, W. L., Guerrero, D. D. S., Melo, M. R. A. (2019). How many abilities can we measure in computational thinking? a study on bebras challenge. In: *Proc. of the 50th ACM Technical Symposium on Computer Science Education*, SIGCSE 19, page 545551, New York, NY, USA. ACM.

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

Barendsen, E., Mannila, L., Demo, B., Grgurina, N., Izu, C., Mirolo, C., Sentance, S., Settle, A., Stupurienė, G. (2015). Concepts in k-9 computer science education. In: *Proc. of the 2015 ITiCSE on Working Group Reports*, ITICSE-WGR 15, page 85116, New York, NY, USA. ACM.

Barr, V., Stephenson, C. (2011). Bringing computational thinking to k-12: What is involved and what is the role of the computer science education community? *ACM Inroads*, 2(1), 4854. 21.

Bell, T., Vahrenhold, J. (2018). Cs unplugged – how is it used, and does it work? In Böckenhauer, H.-J., Komm, D., Unger, W., editors, *Adventures Between Lower Bounds and Higher Altitudes: Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday*, pages 497–521. Springer International Publishing, Cham.

Bellettini, C., Carimati, F., Lonati, V., Macoratti, R., Malchiodi, D., Monga, M., Morpurgo, A. (2018a). A platform for the Italian Bebras. In: *Proc. of CSEDU 2018 – Volume 1*, pages 350–357.

Bellettini, C., Lonati, V., Malchiodi, D., Monga, M., Morpurgo, A. (2018b). Informatics and computational thinking: A teacher professional development proposal based on social-constructivism. In *Proc. of ISSEP 2018*, volume 11169 of *LNCS*, pages 194–205. Springer.

Bellettini, C., Lonati, V., Malchiodi, D., Monga, M., Morpurgo, A., Torelli, M., Zecca, L. (2014). Informatics education in Italian secondary school. *ACM Trans. on Comput. Educ.*, 14(2), 15:1–15:6.

Bellettini, C., Lonati, V., Monga, M., Morpurgo, A., Palazzolo, M. (2019). Situated learning with bebras tasklets. In: *Proc. of ISSEP 2019*, volume 11913 of *LNCS*, pages 225–239. Springer.

Budinská, L., Mayerová, K. (2019). From bebras tasks to lesson plans - graph data structures. In: *Proc. of ISSEP 2019*, volume 11913 of *LNCS*, pages 256–267. Springer. Calcagni, A., Lonati, V., Malchiodi, D., Monga, M., Morpurgo, A. (2017). Promoting computational thinking skills: Would you use this Bebras task? In *ISSEP 2017*, volume 10696 of *LNCS*, pages 102–113. Springer.

Chiazzese, G., Arrigo, M., Chifari, A., Lonati, V., Tosto, C. (2018). Exploring the effect of a robotics laboratory on computational thinking skills in primary school children using the Bebras tasks. In *Proc. of TEEM*, pages 25–30. ACM.

Computing at School (2012). Computer science: A curriculum for schools.

Corradini, I., Lodi, M., Nardelli, E. (2017). Conceptions and misconceptions about computational thinking among Italian primary school teachers. In: *Proc. of ICER 2017*, pages 136–144. ACM.

Dagienė, V., Futschek, G. (2008). Bebras international contest on informatics and computer literacy: Criteria for good tasks. In: *Proc. of ISSEP 2008*, volume 5090 of *LNCS*, pages 19–30. Springer.

Dagienė, V., Futschek, G., Stupurienė, G. (2016). Teachers' constructionist and deconstructionist learning by creating bebras tasks. In: *Proc. of Constructionism 2016*, pages 257–264.

Dagienė, V., Futschek, G., Stupurienė, G. (2019). Creativity in solving short tasks for learning computational thinking. *Constructivist Foundations*, 14(3), 382–396.

Dagienė, V., Mannila, L., Poranen, T., Rolandsson, L., Söderhjelm, P. (2014). Students performance on programming-related tasks in an informatics contest in Finland, Sweden and Lithuania. In: *Proc. of ITiCSE 2014*, pages 153–158. ACM.

Dagienė, V., Sentance, S. (2016). It's computational thinking! Bebras tasks in the curriculum. In: *Proc. of ISSEP 2016*, pages 28–39. Springer.

Datzko, C. (2019). The genesis of a bebras task. In: *Proc. of ISSEP 2019*, volume 11913 of *LNCS*, pages 240–255. Springer.

Dewey, J. (1938). *Experience and education*. Collier Books, New York.

Dong, Y., Cateté, V., Lytle, N., Isvik, A., Barnes, T., Jocius, R., Albert, J., Joshi, D., Robinson, R., Andrews, A. (2019). Infusing computing: Analyzing teacher programming products in k-12 computational thinking professional development. In: *Proc. of ITiCSE 2019*, pages 278–284. ACM.

Forlizzi, L., Lodi, M., Lonati, V., Mirolo, C., Monga, M., Montresor, A., Morpurgo, A., Nardelli, E. (2018). A core informatics curriculum for Italian compulsory education. In: *Proc. of ISSEP 2018*, pages 141–153.

Glasersfeld, E. V. (2001). The radical constructivist view of science. *Foundations of Science*, 6:31–43.

Gujberova, M., Kalas, I. (2013). Designing productive gradations of tasks in primary programming education. In: *Proc. of WiPSCE 2013*, pages 108–117. ACM.

Hubwieser, P., Mühling, A. (2014). Playing PISA with bebras. In: *Proc. of WiPSCE 2014*, pages 128–129. ACM.

Hubwieser, P., Mühling, A. (2015). Investigating the psychometric structure of bebras contest: Towards mesuring computational thinking skills. In: *Proc. of LaTiCE 2015*, pages 62–69. IEEE Computer Society.

Izu, C., Mirolo, C., Settle, A., Mannila, L., Stupurienė, G. (2017). Exploring bebras tasks content and performance: A multinational study. *Informatics in Education*, 16(1), 39–59.

Kalos, I., Tomcsanyiova, M. (2009). Students' attitude to programming in modern informatics. *INFORMÁTICA NA EDUCAÇÃO: Teoria & Prática*, 12:127–135.

Lonati, V., Monga, M., Malchiodi, D., Morpurgo, A. (2017). How presentation affects the difficulty of computational thinking tasks: an IRT analysis. In: *Proc. of the 17th Koli Calling Conference on Computing Education*, pages 60–69. ACM.

Mayring, P. (Klagenfurt, 2014). *Qualitative content analysis: theoretical foundation, basic procedures and software solution*.

McCartney, R. and Tenenberg, J., eds (2014). *ACM Trans. Comput. Educ.*, volume 14(2).

Menekse, M. (2015). Computer science teacher professional development in the united states: a review of studies published between 2004 and 2014. *Computer Science Education*, 25(4), 325–350.

Pohl, W., Hein, H.-W. (2015). Aspects of quality in the presentation of informatics challenge tasks. In: *Local proceedings of ISSEP 2015*, pages 21–22, Ljubljana, Slovenia. Ljubljana University.

Seehorn, D., ed. (2011). *K-12 Computer Science Standards – Revised 2011: The CSTA Standards Task Force*. ACM.

Sentance, S., Csizmadia, A. P. (2017). Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Educ. Inf. Technol.*, 22(2), 469–495.

Sheridan, S., Alkhateeb, B., Mouza, C. (2020). Examining pre-service teachers ability to incorporate computational thinking into lesson plans: A comparison of two digital technologies. In: *Society for Information Technology & Teacher Education International Conference*, pages 95–103. Association for the Advancement of Computing in Education (AACE).

Shulman, L. S. (1986). Those who understand: Knowledge growth in teaching. *Educational Researcher*, 15(2), 4–14.

Skudder, B., Luxton-Reilly, A. (2014). Worked examples in computer science. In: *Proc. of the Sixteenth Australasian Computing Education Conference - Volume 148*, ACE 2014, page 5964, AUS. Australian Computer Society, Inc.

Solitro, U., Pasini, M., Gradi, D. D., Brondino, M. (2017). A preliminary investigation on computational abilities in secondary school. In: *Proc. of ISSEP 2017*, volume 10696 of *LNCS*, pages 169–179. Springer.

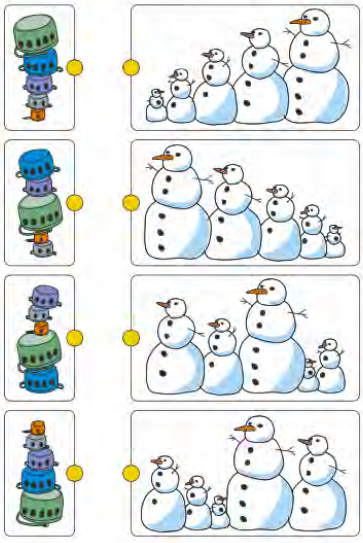
The Royal Society (2012). Shut down or restart? The way forward for computing in UK schools.
The Royal Society (2017). After the reboot: computing education in UK schools.
van der Vegt, W. (2013). Predicting the difficulty level of a Bebras task. *Olympiads in Informatics*, 7:132–139.
van der Vegt, W., Schrivers, E. (2019). Analyzing task difficulty in a bebras contest using cuttle. *Olympiads in Informatics*, 13:145–156.
Vygotsky, L. (1978). *Mind in Society: Development of Higher Psychological Processes*. Cambridge: Harvard University Press.
Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education*, 14(1), 1–16.
Yang, H., Mouza, C., Pan, Y.-C. (2018). Examining pre-service teacher knowledge trajectories of computational thinking through a redesigned educational technology course. In: *Proc. of ICLS 2018*.

V. Lonati is assistant professor in computer science at the University of Milan. Degree in mathematics and PhD in computer science, her research interests concern programming education and the use of socio-constructivist strategies for computing education in primary schools. She is co-founder of the ALaDDIn (the Lab for Didactics and Dissemination on Informatics) research group at Milan university. She is member of the Scientific Committee of the Italian Bebras Challenge and former member of the International Bebras Board.

Apendix
The Bebras Tasks chosen by teachers

Snowmen

Five snowmen are standing in line. From left to right each gets its hat according to its size. The snowmen get the hats from the top, one by one.



Question / Challenge

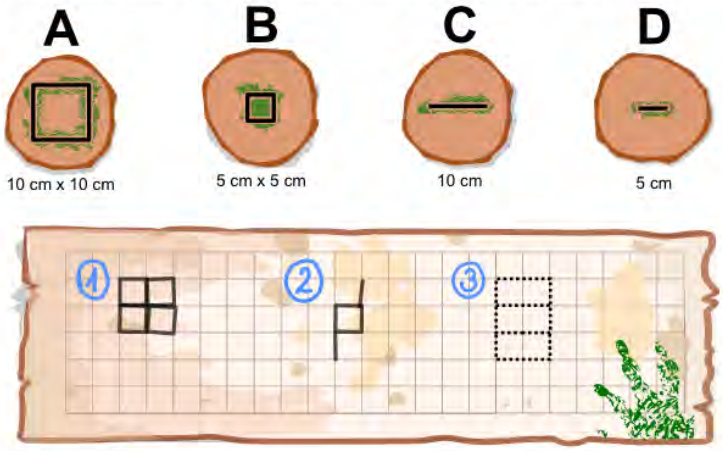
Which pile of hats belongs to which row of snowmen?

Stamps

Beaver Paul has 4 stamps: A, B, C and D in the picture below. By using these, he has already made two figures, 1 and 2, below.

- To create figure 1 Paul used only stamp B (four times).
- To create figure 2 Paul used stamp B (once) and stamp D (twice).

Now Paul wants to make figure 3 below, and his friend Mary offers to help him.



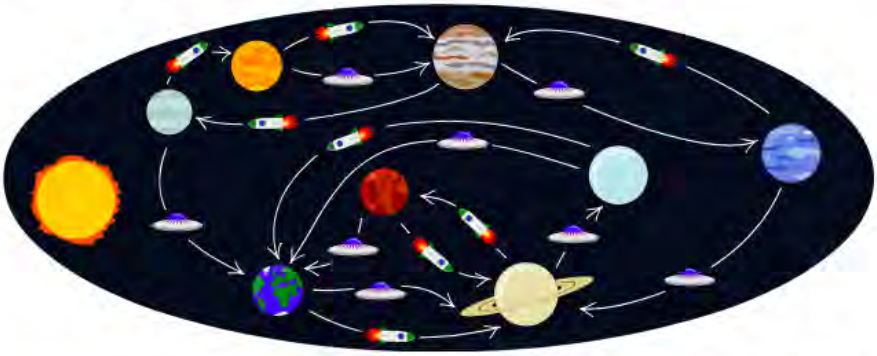
Question / Challenge

Mary claims that she can make figure 3 by using only one stamp twice!

Check the stamp she would use.

Space travel

Astronauts can travel between planets of our solar system with rockets or with UFOs.
The following map shows possible trips:



An astronaut flying from Venus to Saturn can travel with a rocket or with an UFO to Jupiter. From there he can fly with an UFO to Neptune and finally with an UFO to Saturn. So if the astronaut uses a rocket at first and then twice an UFO he describes it as:

The astronaut Heidi is on Neptune and wants to go back to Earth. The Space Travel Agency sends her four suggestions.

Question / Challenge

Which of the suggestions does *not* bring Heidi back to Earth?

Answer Options / Interactivity Description

A)

B)

C)

D)

Cloud communication

A weather beaver sends messages from the top of a mountain to beavers in the valley below. She makes small and large smoke clouds and uses the code below.

			
thunderstorm	light rain	cloudy	sunny

One windy day, the beavers in the valley were only able to see two large clouds as below:

Question / Challenge

Select all messages that might have been sent.

Answer Options / Interactivity Description

A) thunderstorm

B) light rain

C) cloudy

D) sunny

Lockers

There are many lockers installed in an aqua park. To evaluate how they are used, data is collected each minute and appended to a database.

Initially, at 12:30, the data in the database looks like: 1 1 0 0 0 0 0 0 0 1 1 0 (see left picture)

After one minute we will have the following data in the database (see right picture):

1 1 0 0 0 0 0 0 0 1 1 0 1 1 0 0 0 0 0 1 1 1 1 1

After another more minute the data in the database looks like:

1 1 0 0 0 0 0 0 0 1 1 0 1 1 0 0 0 0 0 1 1 1 1 1 1 0 0 0 0 0 0 0 1 1 0

Question / Challenge

How the lockers look like?

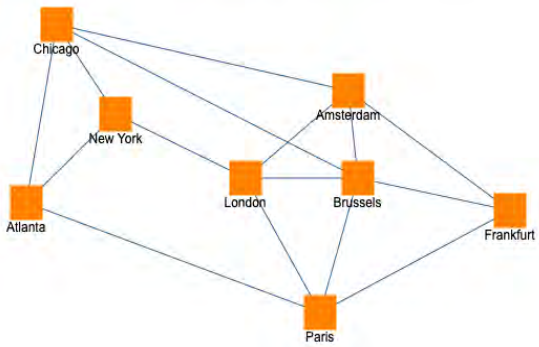
Answers

A) B) C) D)

Greener flight routes

Greener Flight Routes

An international airline has a lot of flight routes connecting several big cities in the world. These cities, and the existing routes between them, are shown on the following picture:



In order to try to decrease its level of CO_2 emissions to help protecting the climate, and therefore our Planet, the company wants to suppress some of the flight routes it proposes. But in order to keep all the cities connected alltogether, the company cannot suppress all the routes.

Given the flight routes map shown above, what is the maximal number of routes the company can suppress so that all the customers can still reach all the cities from any city.

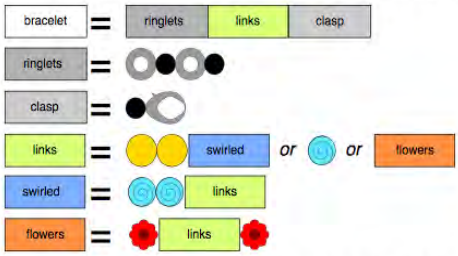
Answers

A. 6
B. 7
C. 8
D. 9

Friendship bracelets

Friendship Bracelets

Stephen likes to make bracelets for his friends, always using the system from the picture below:

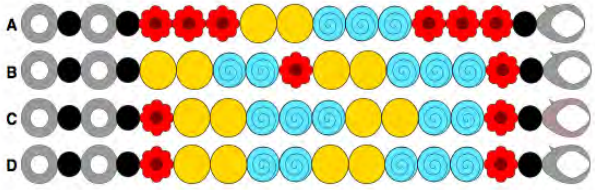


Every bracelet has ringlets, then links and finally a clasp. Here are two examples of what Stephen made.

Stephen made a bracelet for four of his friends, again using the same system. One of the friends broke the bracelet wire and had to restring it, and unfortunately made a mistake.

Which of the four bracelets below is the one with the mistake?

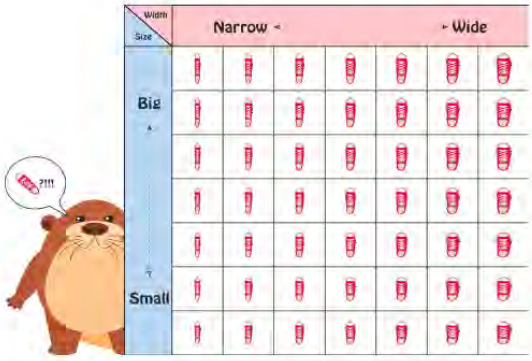
Answers



Buying shoes

Beaver went to the shoe store to buy a pair of shoes. He saw several shoes on display arranged as shown in the picture. The shoes were arranged in increasing order of size as well as width. The shoes varied in size and width, with the smallest and the narrowest shoes kept in left bottom and the biggest and the widest shoe at the top right. All shoes have different size and width.

Being a forgetful beaver, he did not remember his shoe size and will have to try on shoes till he finds the right fit. A right fit is one that is the right size and the right width.



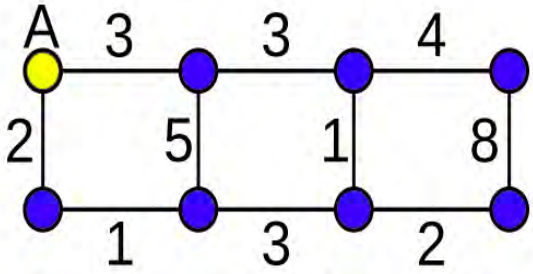
Beaver uses a method that guarantees that he can find the shoe that fits him in 'n' tries.

Question / Challenge

What is the least possible value of 'n'?

Delivery service

You drive for a delivery service called Byber. You start at location A, and you have to drop off packages at the seven other locations, shown as circles. You get paid based on the roads you take, and so, you would like to take the longest trip. You cannot visit any location more than once on your trip. You can finish at any location that you wish. The price you are paid for taking each road is shown below:



What is the most amount of money you can make dropping off these 7 packages?

Answers

A. 23
B. 24
C. 25
D. 26