It's Computational Thinking! Bebras Tasks in the Curriculum

Valentina Dagienė[1] and Sue Sentance[2(✉)]

[1] Vilnius University Institute of Mathematics and Informatics, Akademijos Street 4, 08663 Vilnius, Lithuania
valentina.dagiene@mii.vu.lt
[2] Department of Education and Professional Studies, King's College London, 150 Stamford Street, London SE1 9NH, UK
sue.sentance@kcl.ac.uk

Abstract. Bebras is an award-winning, international contest and challenge in informatics that has been running for 12 years in primary and secondary schools, with 50 countries now participating. From a single contest-focused annual event the Bebras developed to a multifunctional challenge; an activities-based educational community-building network has grown up where the development of Bebras tasks has taken a very significant role. Bebras tasks present a motivating way to introduce computer science concepts to students as well as developing computational thinking skills. Tasks are categorized in terms of the concepts being covered, and each task includes an explanation of how the task relates to informatics. In this paper we propose that Bebras tasks can be used within the school curriculum (whether it is called informatics, computer science, computing or information technology) to promote computational thinking and provide teaching materials. We give examples of Bebras tasks that could be incorporated into the curriculum, and make recommendations for schools wishing to develop children's computational thinking skills.

Keywords: Bebras contest · Computational thinking · Computer science education · Informatics curriculum · Informatics education · Task solving

1 Introduction

There is an increasing focus on computational thinking within the teaching of computer science, computing or informatics (from here on referred to as informatics) in school. Computational thinking was only recently popularised as a concept in 2006 by Wing (2006), although the original definition stems from Papert (1996). Wing claims that computational thinking is for everyone and involves "solving problems, designing systems and understanding human behaviour, by drawing on the concepts fundamental to computer science" (Wing 2006, p. 34). Some new informatics curricula have a significant focus on computational thinking skills being developed, for example in England (Brown et al. 2014) and Poland (Syslo and Kwiatkowska 2015). In the longstanding Bebras contest (Bebras 2016), tasks are designed which demonstrate computer science principles whilst engaging students in problem-solving in a motivating way.

A. Brodnik and F. Tort (Eds.): ISSEP 2016, LNCS 9973, pp. 28–39, 2016.
DOI: 10.1007/978-3-319-46747-4_3

Bebras is an informatics education community-building model and is designed to promote informatics learning in school by solving short concept-based tasks (Dagiene and Stupuriene 2016). Tasks are the most important component of the Bebras model. Each Bebras task should include at least one informatics concept, attract children's attention by a story, picture or interactivity, be short (fits in a computer screen), and not require specific technical knowledge. Some countries use the Bebras to strengthen collaborative learning; for example, in Germany pupils solve Bebras tasks in pairs during a contest and discussions are allowed between the pairs.

Alongside the initial goal of the Bebras project to motivate pupils to be more interested in informatics topics there is a strong intention to deepen algorithmic, logical and operational thinking and, more recently, computational thinking as well. The Bebras challenge intends to promote students' interest in informatics (also in a better understanding of the usage of technology) from the very beginning at school and to motivate students to learn and master technology (Dagiene and Futschek 2008). In the past few years, the number of Bebras challenge participants has been notably growing and exceeded 1.3 million during the Bebras week in November 2015.

In this paper we argue that Bebras is thus a non-formal activity and a possible way in which to incorporate computational thinking into the primary and secondary school curricula, and suggest some exemplar activities to incorporate this.

2 Computational Thinking

The term 'computational thinking' is primarily accredited to Jeanette Wing (Wing 2006), but actually originated with Seymour Papert (Papert 1996). There are differences between these two definitions in that Wing's definition is more focused on problem solving and Papert's definition is more focussed on ideas and analysis (Mannila et al. 2014). Subsequent research has expanded and interpreted the term further (Lu and Fletcher 2009; Grover and Pea 2013; Selby and Woollard 2013).

Computational thinking is not entirely embraced by all; critics suggest that the term is narrowing (Denning 2009) or that computational thinking processes are widespread in other sciences (Hemmendinger 2010). Among other contributions coming from educators, Lee et al. (2011) suggest that we should start from practical examples of what we mean by computational thinking, and identify the terms "abstraction", "automation", and "analysis" as being particularly useful to understand how young pupils can deal with novel problems. Indeed, there is a huge interest in computational thinking as a means of explaining the thinking processes in informatics in school education (K-12); in USA computational thinking underlies the new curricular developments of the Computer Science Teacher Association in USA (CSTA) and Code.org; in England, computational thinking is at the core of a mandatory new Computing curriculum from age 5 until 16 (Department for Education 2013); and Google have launched a teacher development MOOC purely around computational thinking (Google 2016). Attention has turned to the identification of a set of skills that can be seen to comprise a broad definition of computational thinking, and that encompass the logical and problem-solving skills and thought processes that are applied by computer scientists in their work.

The work by Computing At School in the UK defines the five key computational thinking skills used in K-12 as abstraction, decomposition, algorithmic thinking, evaluation and generalisation (Csizmadia et al. 2015). There is also the question of how much computational thinking development is around computer programming and related topics, for example, physical computing (Przybylla and Romeike 2014). Lu and Fletcher 2009 take the view that computational thinking can be separated from programming, and should be taught before programming teaching starts. In addition, Wing's definition of computational thinking includes understanding the consequences of scale, not only for reasons of efficiency but also for economic and social reasons. CSTA in USA adds broader attitudes like the ability to deal with complexity and open-ended problems, tolerance for ambiguity, and ability to work with others to achieve a common goal (ISTE&CSTA 2011).

Computational thinking is explicitly mentioned in some curricular, for example, here in the curriculum in England, referring to pupils aged 7–11: "*Pupils should be taught to: ... Solve problems by decomposing them into smaller parts*" (Department for Education 2013).

3 Computational Thinking and Bebras

One of the drivers of the Bebras community is a shared understanding that learning concepts at an early age is important for a deeper understanding of various informatics topics. The Bebras learning model focuses on informatics concepts by supporting an understanding of computer science phenomena and the development of computational thinking. For the purposes of Bebras we adopt the broad view that computational thinking is a problem-solving process that includes (but is not limited to) the following characteristics (ISTE&CSTA 2011):

- Formulating problems in a way that enables us to use a computer and other tools to help solve them.
- Logic and predicting analytics.
- Data organizing and analysing.
- Representing data through abstractions such as models and simulations.
- Automating solutions through algorithmic thinking (a series of ordered steps).
- Identifying, analysing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources.
- Generalizing and transferring this problem solving process to a wide variety of problems.

One suggested classification of computational thinking skills follows the work of Selby and Woollard (2013) and has been adopted by Computing At School in the UK in developing guidance on computational thinking for teachers (Csizmadia et al. 2015). This describes aspects of computational thinking skills exhibited by students as falling into the five categories below:

1. Abstraction
2. Algorithmic thinking

3. Decomposition
4. Evaluation
5. Generalisation

Based on a previous Bebras categorisation system (Dagiene and Futschek 2008) and further developments with relation to Bebras tasks' content, we can identify the main informatics concept introduced in the task and very broadly divide the content of the task into one of these five areas (categories):

1. Algorithms and programming
2. Data, data structures and representations (includes graphs, data mining)
3. Computer architecture and processes (includes anything to do with how the computer works - scheduling, parallel processing)
4. Communications and networking (includes cryptography, cloud computing)
5. Interaction (Human-Computer Interaction, HCI), systems and society

Analyses of the Bebras tasks used in the 2014 contest were conducted according to the cognitive skills' domains (Bloom taxonomy): this showed that the most tasks demonstrated higher-order thinking skills in the Bloom's taxonomy: Understanding, Applying, Analysing and Evaluating (Dagiene and Stupuriene 2014). In another analysis examining the topics of all Bebras tasks used between 2010 and 2014, the most commonly occurring computational thinking topics were algorithms (66 %) and data representation (38 %), followed by abstraction (16 %) (Barendsen et al. 2015).

In this paper we analyse Bebras tasks that were chosen by Lithuania and UK for all age groups in 2015: in total these amount to 52 tasks, of which the two countries have 35 in common (presented in italics). For each task we allocated the primary and most important computational thinking skill being developed in that task (Table 1), even though we acknowledge that a given task may in some cases develop more than one computational thinking skill.

In Table 1 we can see that of the 52 tasks chosen between the two countries, 22 of them involved some degree of algorithmic thinking in finding a solution. 11 tasks involve the skill of evaluation, 8 demonstrate abstraction, 6 decomposition, and 5 generalisation. Tasks can demonstrate more than one computational thinking skill but in this instance we have highlighted the most dominant one. The emphasis on algorithmic thinking (42 % of tasks) is interesting and supports the observations by Barendsen et al. (2015) about previous tasks. Is it the case that computer scientists use algorithmic thinking more than other computational thinking skills? Or do Bebras task authors find it easier to write tasks that involve either executing, debugging or creating an algorithm? We surmise that it may be a combination of these factors: Bebras tasks are short and designed to be solved within 3 min. It may be difficult to generate tasks that demonstrate a lot of decomposition or evaluation in a short task. However, a key aspect of computer science at school level is the design and execution of algorithms, which supports the development of programming skills, so it may not be surprising that so many algorithmic thinking tasks make their way into the Bebras contest.

Table 1. Bebras 2015 task analysis according to computational thinking (CT) skills

CT Skill	Tasks	Example
Abstraction	*Beaver the Alchemist* Busy Beaver *Drawing Stars* Fried egg *Geocaching* *Popularity* Trains *Walnut Animals*	**Walnut animals:** With walnut animals, we abstract from features like fur and size. We represent the animal only by the structure of its body; the rest is unimportant. This structure is preserved even when the animals are transformed. A computer scientist must recognise what is important, what can be left out, and how structures are similar
Algorithmic thinking	*Beaver Logs* *Biber Hotel* *Bowl Factory* *Building a Chip* Button Game Car Transportation *Chakhokhbili* *Crane operating* *Cross Country* Decorating Chocolate *Drawing Patterns* *Dream Dress* *Fair Share* *Irrigation system* Left Turn! *Mushrooms* Pencils Alignment Reaching the Target *Supper Power* *Family Theatre* *Throw the Dice* *You Won't Find It*	**Biber hotel:** The structure of the beaver hotel is a so-called "binary tree", meaning that from every there are two branches leaving to further rooms. The room number facilitates further navigation. Data on a computer can also be organised in such a way. Despite having several millions of entries, an entry (or its absence) can be found in less than 25 comparisons. In fact, with at most n comparisons it is possible to distinguish between 2^n-1entries **Crane operating:** In this task a sequence of instructions is searched for. Two objects can only be changed if one of the objects is placed at an empty place. Most computers still work with sequentially-run programs, so each exchange operation in the memory of the computer also needs an extra space

(*Continued*)

Table 1. (*Continued*)

CT Skill	Tasks	Example
Decomposition	*Animation* *Fireworks* *Pirate Hunters* *Stack Computer* *Quick Beaver Code* *Word Chains*	**Stack computer:** The usual notation for arithmetic expressions is not the easiest to understand for a computer, or rather, it takes a more complicated program to process such expressions. However writing a program to analyse expressions in postfix notation (or stack computer) is much easier. To solve this task the expression must be broken down (decomposed) into its individual parts
Evaluation	*Animal Competition* *Beaver Gates* *Beaver Tutorials* Birds *Bracelet* *Birthday Balloons* Data Protection *Email Scam* Robot the Stairs Setting the Table Turn the Cards	**Bracelet:** It is important to be able to recognise patterns which may be useful to us. Recognising patterns helps us to find similarities in things that may look different at first, but have something in common. This task also deals with verifying a proposed solution: the possible answers need to be checked against the original bracelet to see if they meet the required order of the shapes
Generalisation	*Beaver Lunch* Kangaroo *Mobiles* RAID Array Spies	**Mobiles:** If you detach a stick (except the uppermost one) from a mobile, you have a mobile again, with the detached stick being the uppermost stick now. That is, the parts of a mobile are constructed in the same way as the full mobile is constructed. If a single figure is considered as a mobile, mobiles may be defined as follows: a mobile is either (a) a single figure, or (b) a stick with one or more mobiles attached to it. In order to define a "mobile", we use the term "mobile" itself. That is a recursive definition, an important concept in computational thinking

4 Bringing Bebras into the Curriculum

As seen above, there is a clear link between Bebras tasks and the development of computational thinking skills, thus demonstrating their potential to be used in the curriculum to develop these skills. In addition, Bebras tasks can be used to demonstrate

specific informatics topics and concepts. In this section, we will illustrate this with some examples of previous Bebras tasks that could be incorporated into an Informatics curriculum in any country. Three curriculum areas have been selected that are currently taught in schools in England and Lithuania, together with some Bebras example tasks are that can be used in school; these areas are: data structures, logical operators and networks.

4.1 Learning About Data Structures

There have been many Bebras tasks in previous years that could be introduced to students which might support an understanding of data structures such as trees, graphs, stacks queues etc. Two examples are discussed below (Figs. 1 and 2).

The structure of the beaver den is a so-called "binary tree", meaning that from every room (a node) there are (possibly) two branches leaving to further rooms. The room-number (or any other ordered data) serves to navigate and find a room again. Data on a computer can also be organised in such way (like for instance names and phone numbers). In fact, with at most n comparisons (depth of the tree) it is possible to distinguish between 2^n-1 entries. For n = 10 we have 1023 possible entries, for n = 20 we have a little over 1 million entries and for n = 30 over one billion.

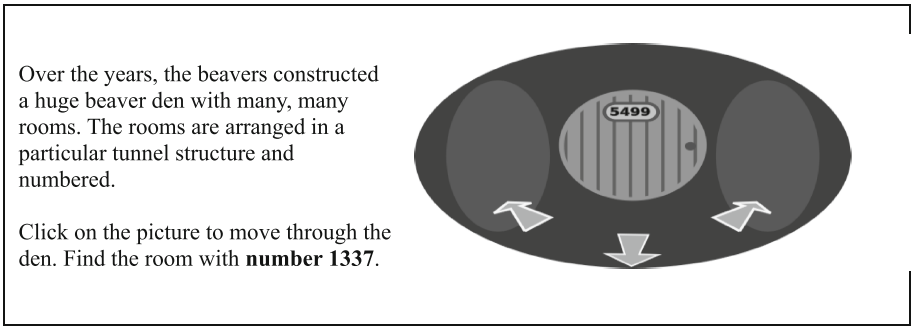


Fig. 1. Biber hotel: a task on a binary tree concept (Ivo Blöchliger, Switzerland)

The Animation task shown in Fig. 2 deals with a data structure concept, in particular that of class, which is very important concept in object oriented programming.

B-taro is planning an animation, which shows a sequence of pictures of a face. The animation should run smoothly. The order of the pictures will be correct if only one attribute of the face changes from one picture to the next. Unfortunately, the pictures got mixed up. Now B-taro must find the correct order again. Luckily, he knows which picture is last. He labels the five other pictures with letters A to E.

In order to find the differences between the pictures, pupils have to find out about the essential attributes of the depicted faces first. The list of attributes and their possible values is: ears: small, large; mouth: plain, smile; nose: small, large; number of teeth: 2, 3; whiskers: curly, straight. For instance, pupils can describe the first face as a list of attribute-value pairs: (ears: small; mouth: plain; nose: large; number of teeth: 3; whiskers: straight).

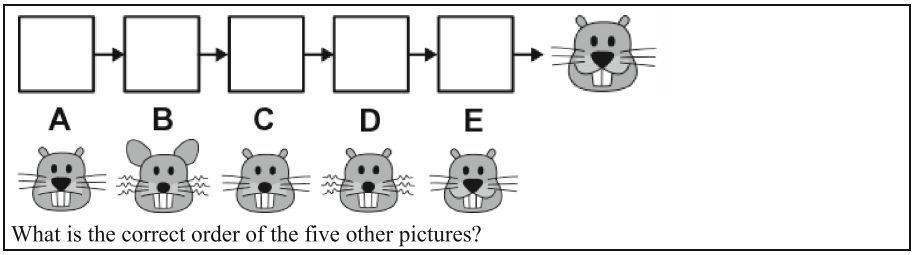


Fig. 2. Animation – a task on a class concept of object-oriented programming (Tomohiro Nishida, Japan))

4.2 Learning About Logical Operations

In many countries, understanding logical operations is a key part of the informatics curriculum. In the national curriculum in England, pupils have to "understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in circuits and programming" at ages 11–14 (Department for Education 2013). Bebras tasks can be focused around different aspects of this topic, particularly tasks where students have to demonstrate an understanding of AND, OR and NOT, or combinations of these operations, in order to solve a task (Fig. 3). The use of such tasks can have a direct applicability to the curriculum.

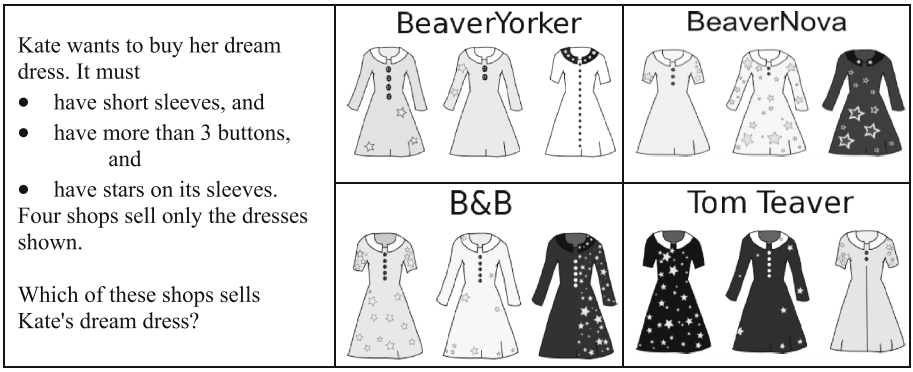


Fig. 3. Dream dress (Karolína Mayerová, Slovakia)

The Dream Dress task involves statements (conditions) that must be evaluated (determined to be true or false) for a set of objects (coats). Conditions and their evaluation is an important part of programming and algorithmic thinking. Conditions can be simple statements. However, more complex statements can be formed using logical operators such as AND, OR, NOT, etc. This task uses the AND operator.

4.3 Learning About Networks

The topic of networks is very broad; it can be found in various forms in many countries' informatics curricular (Barendsen et al. 2015). At school level, this topic could cover topologies, communication, networking protocols, security and the way that the internet is structured. The communication offered by networking can also be seen in examples of social networks, as in the following task (Fig. 4).

A social network is a network used for communication and will be familiar to many students engaging in the Bebras contest. Social networks present us with examples of large and complex networks. It is not always obvious that by posting something on a friend's page, it might be available to people other than the close friend.

Social networks themselves are incredibly powerful tools in today's world. Computing statistics on their users and their pages is useful to marketing departments and anyone else trying to understand a person or group of people. *Instadam c*ould also be interpreted as a model of a miniature internet, with the beavers being websites and friends as pages "linked to". Search engines typically rank these websites by some measure of popularity or importance, at least by the number of links to and from the website. A widely used way to find the result by using a computer is to use the flood fill algorithm which can cope with systems with more than the two iterations in this example.

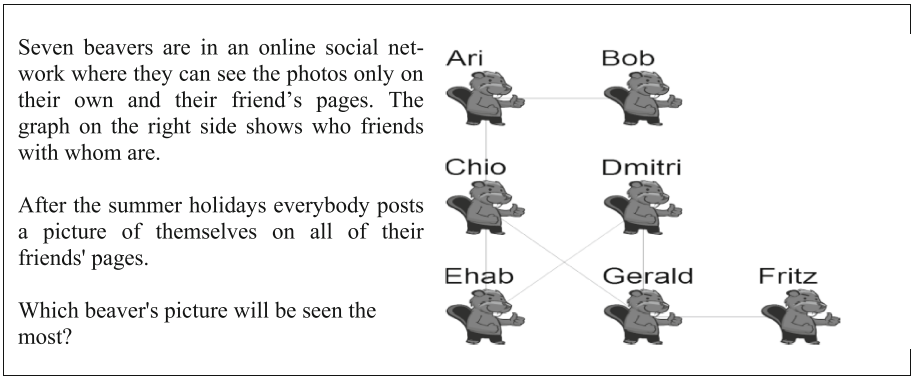
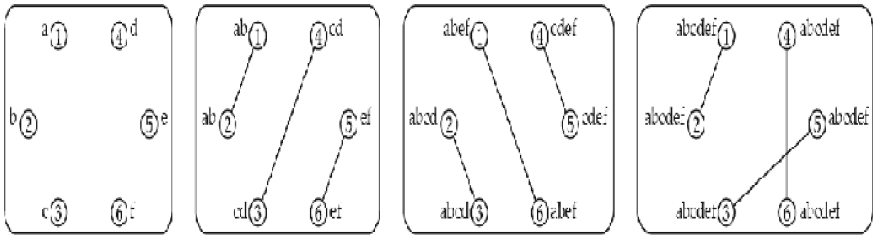


Fig. 4. Popularity (J.P. Pretti, Canada, Cristian Datzko, Switzerland, Sarah Hobson, Australia)

Another key aspect of networks which will be covered in the school curriculum is security. The example Spies (Fig. 5), focusing on spies exchanging information, illustrates a Bebras way of introducing this in school.

These examples illustrate the direct connection from topic to task which can be exploited in the classroom. All examples given here are from the 2015 contest, but as the competition has run since 2004, there are many more examples of tasks that demonstrate computer architecture, principles of operating systems, cryptography and other concepts relevant to the curriculum.

Every Friday, six spies exchange all the information they've gathered in the week. A spy can never be seen with more than one other spy at the same time. So they have to conduct several rounds of meetings where they meet up in pairs and share all information they have at that point. The group of 6 spies needs only three rounds to distribute all secrets: Before the meetings each spy holds a single piece of information. (spy 1 knows 'a', spy 2 knows 'b, etc.). In the first round spies 1 and 2 meet and exchange information so now both know 'ab'. The diagram shows which spies meet in each round with a line. It also shows which pieces of information they all have. After three rounds all information has been distributed.



After an international incident one spy has stopped attending the meetings. What is the minimum number of rounds needed for the five remaining spies to exchange all information?

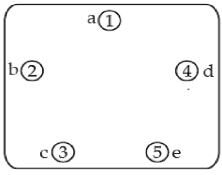


Fig. 5. Spies (Janez Demsar, Slovenia)

5 Pedagogical Issues

The question remains as to the identification of teaching approaches that can draw on Bebras tasks as a resource. To a certain extent the country's curriculum will dictate which tasks are appropriate to be incorporated into a scheme of work. However the tasks lend themselves to being interesting starter tasks for the beginning of a lesson or plenary tasks, for the formative assessment part of a lesson. Currently many teachers use previous tasks as preparation for their students prior to the contest each November; with the growing number of available tasks Bebras tasks could be used in teaching all year round.

Planning lessons around relevant Bebras tasks can only be achieved if Bebras tasks are available and the content is clearly signposted. A new two-dimension categorisation system being proposed for Bebras tasks (Dagiene and Sentance in review) will assist with this. Within this categorisation, each task is classified in terms of its computational thinking skills and informatics concepts. Teachers will be able to use this categorisation to select material for teaching. One situation that can be envisaged is that each country (or countries sharing a common language) has a database of previous tasks that could be searched via concept or computational thinking skill.

Another key area for consideration is assessment. In the Bebras contest, tasks are marked automatically and teachers have access to the final results of their students. By using the tasks for formative assessment in lessons, teachers can track their students' progress in developing computational thinking skills.

6 Conclusion

Bebras tasks present a motivating way to introduce informatics concepts to students as well as developing computational thinking skills. Bebras task developers seek to choose interesting tasks (problems) for enabling students to understand informatics and to think deeper about technology. Moving forward these tasks should cover a range of as many different informatics topics as possible. In addition tasks can be designed which aid the development of core computational thinking skills such as abstraction, algorithmic thinking, decomposition, evaluation and generalisation.

In this paper, the use of Bebras tasks in teaching to promote computational thinking and the introduction of concepts has been suggested through possible examples. Bebras tasks are categorized in terms of the concepts being covered, and can also include a categorisation by computational thinking skill. To support teachers developing lessons, each task includes an explanation of how the task relates to informatics. This can also support teachers who are not fully confident in the subject matter around the tasks, and add to their own professional development. Further work is needed to evaluate the extent to which the use of these tasks in the classroom can support the learning and assessment of computational thinking.

Acknowledgements. The authors thank all members of the international Bebras community who took part in task development and in this way influenced the outcomes of this paper. In addition, we are grateful to Chris Roffey for the development of the UK Bebras Answer Booklet 2015 from which we have taken some ideas for explanation of the example tasks in this paper.

References

Aarts, R.M.: Gossiping. From *MathWorld*–A Wolfram Web Resource. Created by Weisstein, E.W. (2016). http://mathworld.wolfram.com/Gossiping.html. Accessed 30 Apr 2016

Barendsen, E., Manilla, L., Demo, B., Izu, C., Grugina, N., Mirono, C., Sentance, S., Settle, A., Stupuriene, G.: K-9 concepts in computer science education. ITICSE Working Group report (2015)

Bebras International Challenge on Informatics and Computational Thinking. http://www.bebras.org/en/facts. Accessed 30 Apr 2016

Brown, N., Sentance, S., Crick, T., Humphreys, S.: Restart: the resurgence of computer science in UK schools. ACM Trans. Comput. Educ. **14**(2), 9 (2014)

Csizmadia, A., Curzon, P., Dorling, M., Humphreys, S., Ng, T., Selby, C., Woollard, J.: Computational Thinking: A Guide for Teachers (2015). http://computingatschool.org.uk/computationalthinking. Accessed 10 Apr 2016

Dagienė, V., Futschek, G.: Bebras international contest on informatics and computer literacy: criteria for good tasks. In: Mittermeir, R.T., Sysło, M.M. (eds.) ISSEP 2008. LNCS, vol. 5090, pp. 19–30. Springer, Heidelberg (2008)

Dagiene, V., Sentance, S.: Computational thinking and the Bebras challenge: developing a new task categorization system (in review)

Dagiene, V., Stupuriene, G.: Informatics education based on solving attractive tasks through a contest. Commentarii informaticae didacticae **7**, 97–115 (2014)

Dagiene, V., Stupuriene, G.: Bebras - a sustainable community building model for the concept based learning of informatics and computational thinking. Inform. Educ. **15**(1), 25–44 (2016)

Denning, P.J.: Beyond computational thinking. Commun. ACM **52**(6), 28–30 (2009)

Department for Education: The National Curriculum in England: Computing Programmes of Study (2013). https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study. Accessed 30 Apr 2016

Google for Educators: Exploring Computational Thinking (2016). https://www.google.com/edu/resources/programs/exploring-computational-thinking. Accessed 30 Apr 2016

Grover, S., Pea, R.: Using a discourse-intensive pedagogy and Android's App Inventor for introducing computational concepts to middle school students. In: Proceedings of 44th SIGCSE Technical Symposium on Computer Science Education, pp. 723–228. ACM (2013)

Hemmendinger, D.: A plea for modesty. ACM Inroads **1**(2), 4–7 (2010)

ISTE&CSTA (International Society for Technology in Education & the Computer Science Teachers Association): Operational definition of computational thinking for K-12 education (2011). https://csta.acm.org/Curriculum/sub/CurrFiles/CompThinkingFlyer.pdf

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

Lu, J.J., Fletcher, G.H.: Thinking about computational thinking. ACM SIGCSE Bull. **41**(1), 260–264 (2009)

Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., Settle, A.: Computational thinking in K-9 education. In: Proceedings of Working Group Reports of the 2014 on Innovation & Technology in Computer Science Education Conference, ITiCSE-WGR, pp. 1–29. ACM, New York (2014)

Papert, S.: An exploration in the space of mathematics educations. Int. J. Comput. Math. Learn. **1**, 95–123 (1996)

Przybylla, M., Romeike, R.: Physical computing and its scope - towards a constructionist computer science curriculum with physical computing. Inform. Educ. **13**(2), 225–240 (2014)

Selby, C., Woollard, J.: Computational thinking: the developing definition (2013). http://eprints.soton.ac.uk/356481. Accessed 30 Apr 2016

Syslo, M.M., Kwiatkowska, A.B.: Introducing a new computer science curriculum for all school levels in Poland. In: Brodnik, A., Vahrenhold, J. (eds.) ISSEP 2015. LNCS, vol. 9378, pp. 141–154. Springer, Heidelberg (2015). doi:10.1007/978-3-319-25396-1_13

Wing, J.M.: Computational thinking. Commun. ACM **49**(3), 33–35 (2006)