Openness and Creativity in Solving Short Tasks for Learning Computational Thinking

Christian Datzko
Wirtschaftsgymnasium und Wirtschaftsmittelschule Basel, Switzerland • christian/at/datzko.ch

> Abstract • Openness is an aspect of short tasks that can be used as an indicator of whether creativity is a necessity for solving a task efficiently. When designing a task, it is also possible to change it to be more open and thus to require more creativity but then it is likely to require different computational thinking competences.

Openness in the context of tasks

« 1 » I agree with Valentina Dagienė, Gerald Futschek and Gabrielė Stupurienė's claim that children are necessarily creative in their learning process because they need to construct a model of their environment (§17). The term "creativity," however, has a broad and inconsistent range of meanings, as summarized by Hartmut von Hentig (1998). In my commentary, let me narrow down my definition of creativity by requiring "a significant act made to solve a problem" rather than accepting the recreation of given information in one's head also as a form of creativity.

« 2 » When applying this definition of creativity to short tasks for learning computational thinking (see also Dagiene & Sentance 2016), the terms "divergent thinking" and "convergent thinking" coined by J. P. Guilford (Guilford, Wilson & Christensen 1952) help us to understand the challenges when creating such tasks. For creativity in the learning process, both divergent thinking and convergent thinking are necessary. Like the algorithmic design pattern of "branch and bound," divergent thinking allows the student to come up with new approaches and models, while convergent thinking is needed to cut unpromising approaches and models and to restrict the broad range again.

« 3 » On the one hand, tasks should not be restricted to "convergent thinking." Of course, such tasks do have their place: logical reasoning leading to a single correct solution is a central aspect of the computational thinking (CT) skills present in these short tasks. However, other aspects such as abstraction or generalization are part of CT as well.

« 4 » On the other hand, tasks should not be restricted to "divergent thinking" either. Such tasks certainly have their place: creating a model of how a defined situation might work and evaluating its worth for solving a task are also central aspects of CT skills present in these short tasks. Nevertheless, in the end, in order to be able to grade an answer, some well-defined set of correct answers that can be verified automatically must be available.

« 5 » Most tasks are somewhere in an area of tension between a divergence supporting openness and a convergence supporting restriction. This area of tension between openness and restriction is well known to creators of tasks in general. Any task (be it a short task, as in the case of the target article, or longer) can be analyzed in terms of its openness. According to Andreas Büchter and Timo Leuders (2005), there are three aspects of a task that can be either known (✓) or unknown (–): the starting situation, the method with which to solve the task, and the final situation. The resulting eight different types are shown in Table 1.

« 6 » Creativity is found in the more open types of tasks: problems, searches for applications, inverse problems and open situations. To some degree, reasoning tasks where the solving method is missing are also types of tasks where creativity can be found. For the other three types of tasks, creativity is not helpful when solving them. Closed tasks ask for the application of a given method to find an answer (and applying different solving methods or using a different starting situation will be harmful for finding a solution). Inverse tasks are similar in nature except that in the general case, it might take more effort to get the answer. Examples as such are not creative.

Starting Situation	Solving Method	Final Situation	Type of Task	Example
✓	✓	✓	Example	"This is how you find the shortest path on this map…"
✓	✓	–	Closed Task	"Apply Dijkstra's algorithm to find the shortest path on this map."
✓	–	✓	Reasoning Task	"Show that this is the shortest path on this map."
✓	–	–	Problem	"Here's a map, what could you want to know and how?"
–	✓	✓	Inverse Task	"Come up with a map where Dijkstra's algorithm finds this shortest path."
–	✓	–	Search for Application	"Where else could you use Dijkstra's algorithm?"
–	–	✓	Inverse Problem	"Come up with a map that includes this shortest path."
–	–	–	Open Situation	"Build a navigation system."

Table 1 • Classification of tasks (adapted from Büchter & Leuders 2005 and complemented with example tasks for computational thinking).

Openness in short tasks

« 7 » Some of these types of tasks are not suitable for the short tasks that Dagienė, Futschek and Stupurienė are looking for (§24). The authors aim at classroom activities. However, the short tasks are also similar to and in some cases derived from short tasks in the Bebras International Challenge on Informatics and Computational Thinking.

« 8 » In general, a suitable task should at least provide some information so that the student taking part in the challenge can know if the problem has been solved and can move on to the next short problem. In most cases that means that she has selected one of several multiple-choice answers, entered a number into a field and saved it, or has interacted with an interactive pane and selected or created something there. So open situations or searches for applications are too open for the purpose of the target article. In addition, examples are not suitable because they do not challenge the student to actively do something but rather invite her to read and make up her own mind, for which there are no grading criteria. If, after an example, a question to check understanding can be asked, it will no longer be an example but will become part of the starting situation and/or the method for solving.

« 9 » Some of these types of tasks, however, are commonly used. In the 2018 task set from the Swiss Bebras (Datzko, Datzko & Erni 2018), 48.65% of the tasks are closed tasks (40.54%) or inverse tasks (8.11%), and another 48.65% are problems (45.95%) or inverse problems (2.7%). Only one task (constituting the remaining 2.7%) is arguably a reasoning task.

« 10 » Figure 1 shows the task "Lights on!". The task body[1] reads as follows:

> "Seven switches are connected to seven lamps so that every switch controls exactly three lamps. If a switch is pressed, every lamp that is controlled by the switch and is on will be turned off, and every lamp that is controlled by the switch and is off will be turned on. Which buttons does one have to press to end up with all the lights turned on?" (Pohl et al. 2018: 93, my translation)

« 11 » One could argue that the starting situation is known (the state of all lamps) and that the final situation is also known (that all lamps are turned on). That would leave the solution method open, which makes this a reasoning task. However, one could also argue that the final situation is not the final state of the lamps but a correct order of presses of switches making this a problem instead (starting situation is known, solution method and final situation are unknown).

« 12 » Still, this task requires creativity to be solved efficiently: one has to come up with a strategy to control the state of the lamps. There is not just one correct strategy: students can come up with many different ideas and (in the original interactive challenge version) try them and thus maybe solve the problem partially or go back to earlier states and try again. If a student clicks for long enough, she will perhaps find the correct solution "by accident," but then she will probably have wasted some time needed for other short problems.

A comparison of two similar tasks

« 13 » This potential waste of time is even more obvious if one compares two similar tasks: "Lodge No. 29" (Figure 2) and "Bebras Hotel" (Figure 3). The task body of the first task reads as follows:

> "Milo is doing an internship at a lodge. Today he needs to label one of the holiday cottages. Some cottages are already labeled. He starts at cottage #50. From there he is supposed to
> - go left if the new number is smaller than the number of the cottage he's standing in front of,
> - go right if the new number is larger than the number of the cottage he's standing in front of,
> - label the cottage in front of him if the cottage is not yet labeled.
>
> Which cottage must Milo label with the number 29?" (Budinská et al. 2018: 45, my translation)

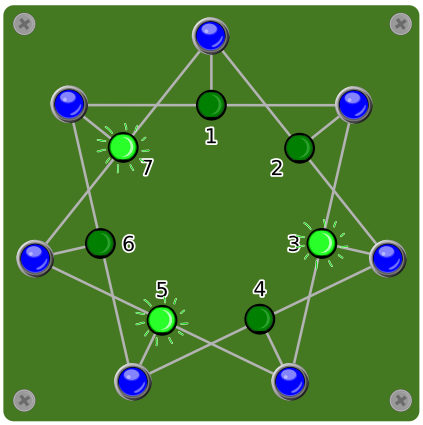

Figure 1 • The interactive pane for the task "Lights on!"

« 14 » And this is the task body of "Bebras Hotel":

> "The beavers rebuilt a big den to create a hotel with many rooms.
> In front of every room passages lead to the left, to the right or back to other rooms. So that the beavers wouldn't get lost, they assigned numbers to the rooms. They used a rule that has to do with going left or right. Because of this rule, rooms that are near each other can have very different numbers. Find room number 1337. If you cannot continue, try going a few steps back and trying again." (Blöchliger, Coolsaet & Pohl 2015: 33, my translation)

« 15 » The explanation of how to solve the task "Bebras Hotel" states that if one wants to find a smaller number than the shown room number, one needs to go left, and if one wants to find a larger number than the shown room number, one needs to go right. This information is not presented to the students as a text, but they are supposed to find out about it by clicking on the arrows and observing the room numbers.

« 16 » Both tasks use a binary search tree and in both tasks a student has to find the place of a specific number. The task "Lodge No. 29" is a closed task because the starting situation is known and the method of how to go from one cottage to the next is also explained. Creativity is not needed to solve this task, but rather strict rule following. The task "Bebras Hotel," however,

1| For the student a Bebras task commonly consists of a "task body," which describes the setting of the task, a "question," which the student has to solve, and in the case of multiple-choice questions, some "answer alternatives," or in the case of interactive tasks, an interactive pane. In addition to that, later on, brochures with an "answer explanation" that explains which answers are correct and which are not, as well as an "It's Informatics" text that connects the task to computer science and CT, are published.

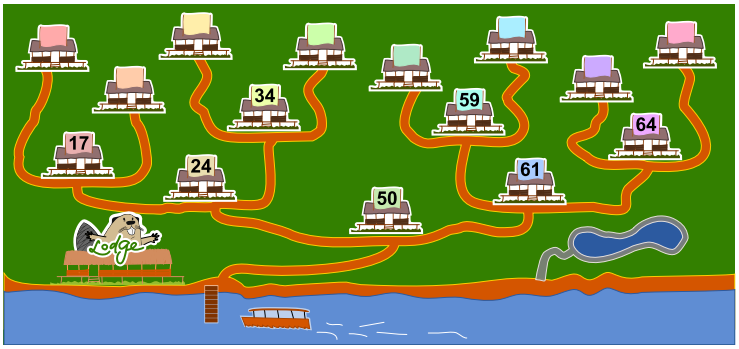

Figure 2 • The graphics for the task "Lodge No. 29."

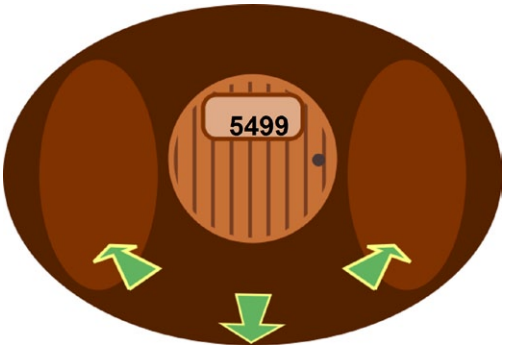

Figure 3 • The interactive pane of the task "Bebras Hotel."

is a problem where the solving method and the final situation are not shown. The student has to create a model of how the rooms could be structured to find room 1337. Only one hint is given by saying that "left" and "right" have a connection to the room number.

« 17 » Both tasks are proper short tasks connected to CT. Although they both look similar, they test different CT skills: "Lodge No. 29" tests the ability of students to follow a (recursive) algorithm, while "Bebras Hotel" requires the student to come up with such an algorithm, which is a much more creative process. Again: the second version could also be solved by trying again and again but it would cost much more time than this task should take, as planned in the challenge.

Conclusion

« 18 » Openness is an aspect of short tasks that can be used as an indicator of whether creativity is a necessity for solving a task efficiently. When designing a task, it is also possible to change it to be more open and thus to require more creativity, but then it is likely to require different CT competences.

« 19 » It looks as if the short problems selected by the authors for their study in primary schools (§§32–51) are all problems rather than closed tasks, although they are both present in approximately equal amounts in the Bebras challenge. This is a wise decision to foster creativity.

« 20 » A big opportunity that is only slowly being exploited is that the wealth of short tasks in the Bebras project, which underwent a rigorous and time-intensive preparation process, offers a treasure chest of tasks for teachers to use in class.[2] Not only can they use them as they are (be it as openers for learning situations, as a new collection of tasks for a specific topic or as a chance for internal differentiation; the target article provides good examples of this in §§41–51), they can also be adapted for further needs. A short task designed for 3–5 minutes could very well be adapted to fill a complete lesson. Büchter and Leuders (2005) describe how to "open" tasks and adapt them for totally different purposes and time frames. By "opening" it, a task can also be made suitable for fostering creativity.

« 21 » An open issue is how metacognition can be cultivated in short tasks, since higher amounts of self-observation and self-evaluation lead to a better and quicker output (Sjuts 1999: 42–44). Future research could explore methods for how to systematically promote metacognition for students solving short tasks as described in the target article and also how to measure the extent to which metacognition has taken place.

« 22 » To conclude, Dagienė, Futschek and Stupurienė show an interesting and promising adaptation of good practices from the Bebras challenge to regular classes for CT in the context of fostering creativity. To continue this line of investigation further development of this method, such as exploring the issue of the openness of a task discussed in this commentary or the proposed research on how to systematically promote metacognition could prove very fruitful.

2| For this reason Switzerland, like many other countries, offers brochures for teachers and students with the tasks from former Bebras challenges to reuse.

References

Blöchliger I., Coolsaet K. & Pohl W. (2015) Biber-Hotel. In: Blöchliger I., Datzko C. & Erni H. (2015) Aufgaben und Lösungen 2015. SVIA-SSIE-SSII Schweizerischer Verein für Informatik in der Ausbildung, Zürich: 33–34. https://www.informatik-biber.ch/documents/2015/Informatik-Biber-2015-Alle-Stufen-mitLoesungen.pdf

Büchter A. & Leuders T. (2005) Mathematikaufgaben selbst entwickeln: Lernen fördern – Leistung überprüfen. Cornelsen Verlag Scriptor, Berlin.

Budinská L., Mayerová K., Tomcsányi P., Moodleah S., Datzko C., Datzko S., Schlüter K. & Erni H. (2018) Ferienhaus Nr. 29. In: Datzko C., Datzko S. & Erni H. (eds.) Aufgaben und Lösungen 2018 – Alle Stufen. SVIA-SSIE-SSII Schweizerischer Verein für Informatik in der Ausbildung, Zurich: 45–46.

Dagienė V. & Sentance S. (2016) It's computational thinking! Bebras Tasks in the Curriculum. In: Proceedings of the international conference on informatics in schools: Situation, evolution, and perspectives. Springer, Cham: 28–39.

Datzko C., Datzko S. & Erni H. (2018) Aufgaben und Lösungen 2018 – Alle Stufen. SVIA-

SSIE-SSII Schweizerischer Verein für Informatik in der Ausbildung, Zürich. https://www.informatik-biber.ch/documents/2018/Informatik-Biber-2018-Alle-Stufen-mit-Loesungen.pdf

Guilford J. P., Wilson R. C. & Christensen P. R. (1952) A factor-analytic study of creative thinking II: Administration of tests and analysis of results. Psychology Laboratory Report 8. University of Southern California, Los Angeles.

Hentig H. von (1998) Kreativität: Hohe Erwartungen an einen schwachen Begriff. Carl Hanser, Munich.

Pohl W., Prantsoudi S., Kelevedjiev E., Monga M., Datzko C., Datzko S. & Weigend M. (2018) Licht an! In: Datzko C., Datzko S. & Erni H. (eds.) Aufgaben und Lösungen 2018 – Alle Stufen. SVIA-SSIE-SSII Schweizerischer Verein für Informatik in der Ausbildung, Zürich: 93–95. https://www.informatik-biber.ch/documents/2018/Informatik-Biber-2018-Alle-Stufen-mit-Loesungen.pdf

Sjuts J. (1999) Mathematik als Werkzeug zur Wissensrepräsentation: Theoretische Einordnung, konzeptionelle Abgrenzung und interpretative Auswertung eines kognitions- und konstruktivismustheoriegeleiteten Mathematikunterrichts. Schriftenreihe des Forschungsinstituts für Mathematikdidaktik Nr. 35, Osnabrück.

Christian Datzko is teaching high-school students mathematics and computer science at the Wirtschaftsgymnasium und Wirtschaftsmittelschule in Basel. He is involved in the Bebras International Challenge on Informatics and Computational Thinking and, amongst other services, is responsible for the tasks of the challenge in Switzerland. His research interest is to deliver high-quality tasks to students that enable them to become interested in various aspects of computer science and to teach them basic computational thinking competences through these tasks. As a member of the Research Department of Music & Media Technology of the University of Osnabrück, he is also researching music information retrieval.

Received: 19 June 2019
Accepted: 24 June 2019

Developing Computational Thinking, "Fad" or "Fundamental"?

Arnold Neville Pears
KTH Royal Institute of Technology, Sweden • pears/at/kth.se

> Abstract • Publicised by Wing and later expanded on, computational thinking purports to be the foundation of thought for coming generations, an indispensable skill-set that compulsory education must provide. The target article uses small tasks to develop skills relevant to computational science and computing, and explores the student interaction with these tasks. Useful skills may be developed by these students, but the article does not explicitly connect these tasks to computational thinking skills or competencies. This causes the reader to ask the question: are they developing computational thinking, or some other set of computer-related skills? A more fundamental question, and one beyond the scope of the target article, is what are the skills that are ultimately unique for computational thinking?

Why comment on computational thinking?

« 1 » This commentary explores the underlying foundations of computational thinking (CT) and the international movement driving the CT agenda. In doing so, two themes are used to focus the debate.

a Computational thinking "fad" or "fundamental"?

b Who and what defines computational thinking?

« 2 » Computation is not new. The notion of using "computers" to perform routine operations in order to solve large-scale numerical problems pre-dates the notion of automating the process with machinery. Early examples of "computers" can be found in the history of the navigation tables of Great Britain, and in the collections of logarithms, sines and cosines and other mathematical reference tables long used as an aid in lengthy calculations requiring numerical solutions. To compile such tables was a herculean task, requiring concentrated planning and considerable effort, not to mention an enormous number of "computers" and "comparators" who would perform the computations and compare the results prior to inclusion in the final book of tables (Grier 2001).

« 3 » These "computers" were often clergy or other mathematically educated and reliable persons, and their collaboration around the systematic compilation of navigational and other types of mathematical table cannot be underestimated (Robson & Stedall 2008). However, the process was fraught with error. Individuals made mistakes, and it was not uncommon for a comparator to either receive two erroneous computations to compare, or to make a mistake in either the comparison or transcription. Thus emerged an urgent priority, to mechanise (automate) these computations to eliminate the element of human fallibility.

« 4 » As automation of computation advanced, the focus in computation shifted from coordinating people to coordinating the automation itself. Early engines provided support for computation (Tedre 2014), but also required regular maintenance and input from human operators. For instance, the mechanical nature of early calculating machinery required precise calibration and lubrication in order to avoid slippage in linkages, of gravel or dust in the works, leading to errors in the results. Examples of programmable mechanical devices formed the basis of some of the long-lasting economic impacts of the first industrial revolution, see, for instance, the impact of knitting machines on the textiles and clothing trades (O'Rourke, Rahman & Taylor 2013).

« 5 » As computation moves beyond specialised areas of application to form the basis for automation of many processes in modern society, the potential for a second technological revolution emerges. Automation is on the cusp of reaching such a level of sophistication that many activities hitherto considered the province of human agency will almost inevitably be automated in the near future. Examples include e-judges in courts of law and self-driving vehicles, to name but two. In this context computational thinking has been advanced as a necessary skill.

« 6 » What equips future generations with the sophisticated awareness and tech-