

Digital games and tools for development of computational thinking in primary school

Martina Holenko Dlab, Natasa Hoic-Bozic, Matea Andelic, Ivica Boticki
University of Rijeka, University of Rijeka, University of Rijeka, University of Zagreb
mholenko@inf.uniri.hr, natasah@inf.uniri.hr, mateaa.andelic@gmail.com, ivica.boticki@fer.hr

Abstract — *Computational thinking is the process that involves formulating a problem and expressing its solution in a way that a human or a machine can effectively perform. Computational thinking include skills that are useful for a career in almost every sector and should be encouraged from primary school. Educational games can motivate students to actively participate in learning activities and have a potential to support development of computational thinking as well as programming skills. Games can be integrated in different school subjects as unplugged activities (without the use of technology) or in a digital form, which is more appropriate for today's students who are growing up in the digital age. This paper analyses the potential of using digital games and tools for supporting the development of computational thinking and programming skills and gives overview of examples that are freely available and suitable for primary school students. In addition, the paper describes a model for development of computational thinking skills designed within the project GLAT that will be further developed as one of the aims of the new project "Digital games".*

Index Terms—Digital games, computational thinking, learning programming, primary education, project GLAT.

I. INTRODUCTION

Computational thinking is considered as one of the fundamental skills for the 21st century, needed to work in the modern business world. Employees that will take an active role in the problem-solving and decision-making are required in all sectors [1], [2].

There are many definitions of the *computational thinking*. Computational thinking is used to solve the complex problem in algorithmic way and to present the solution so that a computer and a human can understand it [1]. In definitions of the term computational thinking, authors usually list the set of skills needed for that process. Among fundamental skills are [2]: decomposition, abstraction, algorithms, debugging, iteration and generalization. Using these skills one can logically organize and analyse data, break the problem into smaller parts, identify and apply existing solutions in order to design efficient solution of the problem, etc. [3], [4]. Mentioned skills are useful for a career in almost every sector, including business and financial markets, tourism, energy, healthcare, education, etc.

Many researchers are dealing with the question *how* these skills can be developed and through what technological means. The development of many skills, including computational thinking and programming skills, can be supported by integrating game-based learning (GBL) strategies [5], [6], and gamification into education [4], [7].

Game-based learning refers to approach where games with defined learning outcomes are used to enhance the learning experience while in gamification game elements are added to a non-game situations [8]. It is believed that playing games is a natural way of learning for a child. Through games, children learn and gain skills but they are not aware of it. There are many benefits of using games in education. Educational games stimulate imagination and creativity, enhance concentration, and improve memory. In addition, by playing games students can gains skills for problem solving, logical reasoning, strategic thinking, and management. Today's children are growing up surrounded by digital technology, which affects their way of thinking and processing information. Therefore, they are more interested in digital games with rich visual elements that attract them into the fantasy world and allow them to forget they are actually learning. Digital games enable students to adapt to new technologies as well as to improve their attention span and spatial vision [6], [9].

Research in the field also deals with the question *when* to introduce the concepts of computational thinking. Studies has shown that there is no need to wait until students are in college [10] since students can start to adopt these concepts in lower grades of primary school [11], especially when games support them in that process [4], [12]. It is important to educate primary school teachers and encourage them to plan learning scenarios that will include games in different school subjects [13]. The precondition for this is that teachers understand the problem solving process as algorithmic process that can be applied to other settings and used to solve problems in similar situations [10].

It should be stressed out that *computational thinking* is not the same as *programming*. Students should have computational thinking skills to develop a solution of the problem and programming skills to tell a computer what to do and how to do it [3]. However, some programming techniques are also used in the process of problem solving (e.g. iterations). Given the relation between computational thinking and programming, game-based educational environments that support the acquisition of computational thinking skills can include different types of programming tasks, and therefore require the knowledge of programming concepts and/or the usage of programming languages. When such games are intended for primary school students, players are expected to know basic programming concepts (such as sequence, loop, variable, conditionals [14]) and usually need to solve problems (tasks) using block-based instructions or programming languages like Scratch and Blockly [14], [15].

Alternative approach to support the learning of basic programming concepts and the development of computational thinking is with unplugged activities [16]. In such activities students do not use digital devices but physical objects (e.g. board games, cards, strings) or movements (e.g. footsteps, dance) to represent and understand concepts. Thus, unplugged approach is convenient for classrooms without the technology infrastructure [17], [18].

The aim of this paper is to analyse the potential of using digital games and tools for supporting the development of computational thinking and programming skills in primary schools. The paper presents a set of games and tools that are freely available on the Web and can be used for this purpose. The research focuses on games and tools suitable for students in lower grades of primary school. Therefore, only basic computational thinking skills (i.e. decomposition, pattern recognition, abstraction, algorithm development [3]) and basic programming concepts (i.e. sequence, loop, variable, conditionals) are included in the analysis.

The paper also presents the goals and activities of the two projects that encourage the integration of computational thinking into the daily teaching of different subjects in primary school using games. These are the project “Games for Learning Algorithmic Thinking (GLAT)”, funded by the Erasmus+ Programme of the European Union under the Key Action 2: Cooperation for innovation and the exchange of good practices - Strategic Partnerships for school education, and the project “Digital games in the context of learning, teaching and promoting inclusive education (Digital Games)”, funded by the University of Rijeka. The paper is organized as follows: Section II gives overview of examples of digital games and tools for development of computational thinking skills while Section III focuses on examples that can be used for acquiring basic programming concepts. Section IV presents a model for development of computational thinking using games in primary school. Section V brings conclusions and plans for future work.

II. DIGITAL GAMES AND TOOLS FOR DEVELOPMENT OF COMPUTATIONAL THINKING SKILLS

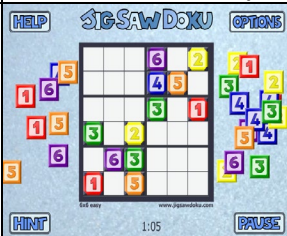
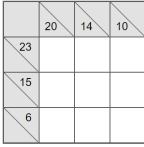

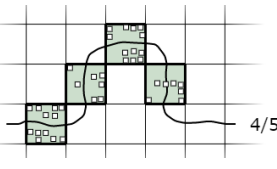

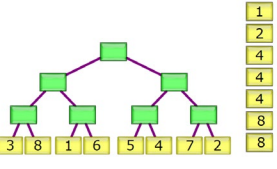
As mentioned before, computational thinking skills which can be developed in primary school students are [3]: decomposition, pattern recognition, abstraction, and algorithm design. By using these skills, one can understand complex problems, develop possible solutions and present these solutions in a way that computers can understand them. Each skill is as important as the other three and if one of them is not applied, the problem cannot be solved successfully.

Decomposition is used to break down a complex problem into smaller problems. In the games that support the development of computational thinking skills, students should recognize smaller problems which are easier to understand, solve them separately, and integrate them in the final (complete) solution. While solving smaller problems, students are expected to recognize patterns (i.e.

to notice similarities among and within problems). *Pattern recognition* enables them to take advantage of previous knowledge, achievements, or experience and reach a solution faster. In addition, during the process of solving the problem, students should focus only on the important details and ignore all irrelevant information. This process is called *abstraction*. To apply abstraction, students need to know how to choose details that can be ignored in order to make the problem easier, without neglecting the important information. When the similar problem needs to be solved, an *algorithm* can be designed. By designing the algorithm, students specify steps or rules to solve each of the smaller problems and these steps can be reused whenever needed.

Table 1 shows examples of games and tools for development of all computational skills mentioned above. The most of these games include tasks of various difficulty and therefore can be used in different grades of primary school. All games are freely available on the Web.

Table 1 - Examples of games and tools for development of computational skills

Game with description/rules	Screenshot of task example
Sudoku Jig Saw Doku [19] – The player should fill all empty squares in a grid of 81 squares which is divided into nine blocks. Each of the nine blocks has to contain the numbers 1 to 9 within its squares. Each number can only appear once in a row, a column or a box.	
Kakuro [20] – The player should fill all empty squares using numbers 1 to 9 so the sum of numbers in each row equals the clue on its left, and the sum of numbers in each column the clue on its top.	
Code Combat [21] – The player should write a program to move the main character around a dungeon and perform assigned tasks (e.g. collect all the gems, don't run into spikes, don't let ogre see him).	
Initial simulation [22] – The player should place towns of required size on river(s). Two towns cannot touch horizontally or vertically. Sometimes, additional conditions are given (e.g. town cannot have tiles on more than 1 river).	
Thinking Myself [23] – The player should solve various tasks. Example: Copy the given picture of a fox by clicking on triangles in the square.	
Bebras Challenge [24] – The player should solve various problems. Example: Beaver used the board and numbered cards to represent winners of each stage on a tournament of races. The runners wore the same numbers throughout the tournament. The task is to put the cards that were removed from the board back to the right places.	

III. DIGITAL GAMES AND TOOLS FOR LEARNING BASIC PROGRAMMING CONCEPTS

There are many games and tools that can be used in primary schools to support the learning of the basic concepts needed for programming: sequence, loop, variable, conditionals [14], [15].

One of the basic principles in programming is that a certain activity or task should be expressed as a *sequence* of instructions that will be executed by the computer. In the games that support the learning of this concept, students are challenged to specify a set of instructions in the appropriate order to solve a given task. By playing such games, the students have the opportunity to observe that different sequences of instructions will give different results.

To use *loops* while playing games, students need to recognize which instruction or a set of instructions is repeated in the task solution as well as a number of repetitions (iterations). Students have the opportunity to detect that the solution which includes one or more loops has significantly less instructions than the one without loops.

Games can also support students in learning how to store data using *variables*. While solving tasks, students usually have to store, retrieve and update values of variables. In games for early primary school, the students are not familiar with the different types of data but use only variables that represent integer numbers (e.g. number of collected items) or text (e.g. the text that will the main character say).

Variables are often used in *conditionals* that enable making decisions based on certain conditions. In games that support learning this concept, students usually need to recognize which conditions could be used to trigger some events and/or direct the flow of the game.

Table 2 shows examples of games and tools that are freely available on the Web and can be used for learning the basic programming concepts. These games and tools are appropriate for primary school students who are starting with learning programming - instead of writing a code, students are programming using drag and drop code blocks. Figure 1 shows an example of using blocks to solve one task in the game "Bee" [25]: use 'repeat' block to collect all of the nectar and make all of the honey. The illustration given with the task is shown on the left side while the sequence of instructions that represents solution of the task is shown on the right side of the figure.



Figure 1 - Example task and its solution from the game 'Bee'

Most of the games from Table 2 support learning of all four concepts so the game that will be used can be chosen according to student's interests (e.g. depending on the students' preferences regarding the main character in the game). Also, several games can be combined to maintain the motivation for learning.

IV. GLAT MODEL FOR DEVELOPMENT OF COMPUTATIONAL THINKING SKILLS USING GAMES IN PRIMARY SCHOOL

The project GLAT [26] encourages the integration of activities for development of algorithmic and computational thinking skills into the daily teaching of different subjects in primary school (from the first to fourth grade) through organization of professional development training for primary junior grade teachers.

The training course for focus group of Croatian teachers was organized using the blended model of e-learning [27] so *face-to-face* workshops were combined with online mentoring in learning management system.

Throughout the course, teachers were introduced to innovative teaching methods including Game Based Learning (GBL), Problem Based Learning (PBL), and Inquiry Based Learning (IBL). Besides theoretical topics, examples of games and digital tools that can be used for development of computational thinking were presented and analysed.

Figure 2 shows the activities for teachers who participated in the training. Participants attended the following two-day workshops: 1) GBL and unplugged activities, 2) PBL, online quizzes and logical tasks, 3) IBL, games and tools for learning programming.

Digital tools presented during the workshop were chosen in accordance with teachers' skills for using ICT. They were introduced to simple digital tools available on the Web which can be used to prepare materials for unplugged activities (1st workshop) or create online quizzes and worksheets with logical tasks (2nd workshop). Unlike teachers of informatics or computer science, primary junior grade teachers are not familiar with programming concepts and do not even know how to use the visual programming languages to create digital games. Therefore, during the third workshop, the basic programming concepts were explained and examples of ready-made games and tasks (shown in Tables 1 and 2) were presented.

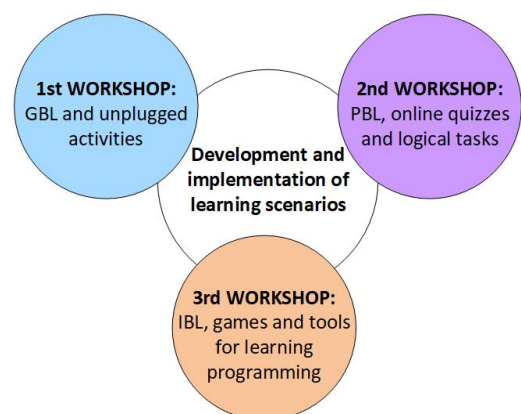
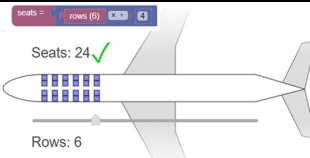

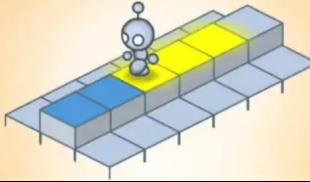


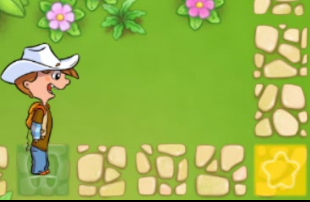



Figure 2 – GLAT workshops

Table 2 – Examples of games and tools for learning the basic programming concepts

Game	Screenshot of task example	Description	Sequence	Loops	Variables	Conditionals
Blockly Demo – Plane seats calculator [28]		The player should build a formula that calculates the total number of seats on the airplane as the number of rows change.	-	-	+	-
Code with Anna and Elsa [29]		The player should use instructions to move Ana or Elsa on ice and create various shapes.	+	+	+	-
LightBot [30]		The player should use instructions to move the blue or pink robot around 3D maze and turn on lights on blue fields.	+	+	-	+
Dragon Dash [31]		The player should use instructions to lead the dragon to coins and treasure by skipping obstacles and foes.	+	+	+	+
Robo Garden [31]		The player should use instructions to lead the robot to ornaments by skipping obstacles and respecting given conditions (e.g. the color of the fields on which the robot can step on).	+	+	+	+
Run Marco! [32]		The player should use instructions to lead the main character (Marko or Sophia) around the given path. On his/her way to the yellow star, the character should skip obstacles and collect gems.	+	+	+	+
Bee (Code.org) [33]–[35]		The player should use instructions to move the bee to the flowers, collect given amount of nectars, then move the bee to the honeycomb and make honey.	+	+	+	+

After each workshop, teachers were encouraged to apply acquired knowledge and skills in development of learning scenarios (i.e. preparations for classes in digital form [36]). They were also expected to implement the developed scenarios in classes with their students. With the help of experts from the project team, teachers developed learning scenarios for different school subjects with educational activities for development of algorithmic and computational thinking [37]. Depending on the workshop topic, games and tasks that teachers included in their learning scenarios were unplugged or supported with appropriate digital tools.

Although many of the teachers were using some of the presented types of games and tasks in their everyday teaching practice even before the training (especially tasks

like puzzles and mazes), experts from the project team help them to understand the process of solving these tasks as algorithmic process. The connection of particular types of games and tasks with computational thinking skills was also explained, which help the teachers in developing their own innovative ideas for using unplugged activities, logical tasks and games that promote algorithmic and computational thinking in different school subjects.

V. CONCLUSIONS AND FUTURE PLANS

To many students playing digital games have become the most common activity in their free time, which gives an opportunity for using digital games for educational purposes as well. By playing games, students can start to learn basic programming concepts and develop

computational skills in early primary school. By integrating games into educational activities, it is possible to improve students' attitudes towards programming and the development of computational thinking, and, in the long term, increase their interest in the selection of future career in the ICT and STEM areas.

Examples of games presented in this paper are suitable for students with different interests (e.g. girls who are more interested in Disney princesses or boys who are more interested in robots) and can be used on different devices (desktop computers, tablets, or smartphones). If they know that these games exist and have the appropriate technology in the classroom, teachers can integrate them in the educational activities in various school subjects.

As a result of the GLAT project, the activities for developing computational thinking and integrating educational digital games into daily teaching from the first grade of primary school will be continuously promoted. In addition, activities of the project "Digital games" which started in 2019 will further explore the possibilities of using educational digital games to improve the quality of teaching and learning through development and promotion of contemporary pedagogical-technological frameworks for the use of GBL. The development of the frameworks will include the selection and development of games and digital tools as well as modern teaching models. It will also include the design of learning scenarios based on developed teaching models applicable in practice for learning and teaching subjects in primary schools.

ACKNOWLEDGMENT

The research has been co-funded by the Erasmus+ Programme of the European Union under the project „Games for Learning Algorithmic Thinking“ (2017-1-HR01-KA201-035362) and by University of Rijeka (Croatia) under the project "Digital games in the context of learning, teaching, and promoting inclusive education" (uniri-drustv-18-130).

REFERENCES

[1] J. M. Wing, "Computational thinking," *Commun. ACM*, vol. 49, no. 3, pp. 33–35, 2006.

[2] V. J. Shute, C. Sun, and J. Asbell-Clarke, "Demystifying computational thinking," *Educ. Res. Rev.*, vol. 22, pp. 142–158, Nov. 2017.

[3] BBC Bitesize, "Introduction to computational thinking." [Online]. Available: <https://www.bbc.com/bitesize/guides/zp92mp3/revision/1>. [Accessed: 16-Mar-2019].

[4] I. Botički, D. Pivalica, and P. Seow, "The Use of Computational Thinking Concepts in Early Primary School," in *Proceedings of the International Conference on Computational Thinking Education*, 2018, p. 6.

[5] M. A. Miljanovic and J. S. Bradbury, "A Review of Serious Games for Programming," in *Serious Games. JCSG 2018. Lecture Notes in Computer Science*, vol. 11243, Göbel S. et al., Ed. Springer, Cham, 2018, pp. 204–216.

[6] M. Zapušek and J. Rugelj, "Learning programming with serious games," *EAI Endorsed Trans. Game Based Learn.*, vol. 13, no. 1, pp. 1–6, 2013.

[7] T. Jagušt, I. Botički, and H.-J. So, "Examining competitive, collaborative and adaptive gamification in young learners' math learning," *Comput. Educ.*, vol. 125, pp. 444–457, Oct. 2018.

[8] R. Al-Azawi, F. Al-Faliti, and M. Al-Blushi, "Educational gamification vs. game based learning: Comparative study," *Int. J. Innov. Manag. Technol.*, vol. 7, no. 4, pp. 132–136, 2016.

[9] Gamelearn, "5 great benefits of game-based learning in soft skills training," 2015. [Online]. Available: <https://www.game-learn.com/game-based-learning-in-soft-s-kills-training-5-great-benefits/>. [Accessed: 25-Mar-2019].

[10] V. Barr and C. Stephenson, "Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?," *Inroads*, vol. 2, no. 1, pp. 48–54, 2011.

[11] W. J. Rijke, L. Bollen, T. H. Eysink, and J. L. Tolboom, "Computational Thinking in Primary School: An Examination of Abstraction and Decomposition in Different Age Groups," *Informatics Educ.*, vol. 17, no. 1, p. 77, 2018.

[12] I. Botički, P. Kovačević, D. Pivalica, and P. Seow, "Identifying Patterns in Computational Thinking Problem Solving in Early Primary Education," in *Proceedings of the 26th International Conference on Computers in Education*, 2018, p. 6.

[13] R. F. Adler and H. Kim, "Enhancing future K-8 teachers' computational thinking skills through modeling and simulations," *Educ. Inf. Technol.*, vol. 23, no. 4, pp. 1501–1514, Jul. 2018.

[14] K. Brennan and M. Resnick, "New frameworks for studying and assessing the development of computational thinking," in *Proceedings of the 2012 annual meeting of the American Educational Research Association*, 2012, p. 25.

[15] A. Bauer, E. Butler, and Z. Popovic, "Approaches for teaching computational thinking strategies in an educational game: A position paper," in *IEEE Blocks and Beyond Workshop (Blocks and Beyond)*, 2015, pp. 121–123.

[16] T. Jagušt, A. Sovic Krzic, G. Gledec, M. Grgić, and I. Bojic, "Exploring Different Unplugged Game-like Activities for Teaching Computational Thinking," in *IEEE Frontiers in Education Conference 2018*, 2018, pp. 1–5.

[17] C. P. Brackmann, M. Román-González, G. Robles, J. Moreno-León, A. Casali, and D. Barone, "Development of Computational Thinking Skills through Unplugged Activities in Primary School," in *Proceedings of the 12th Workshop on Primary and Secondary Computing Education - WiPSCE '17*, 2017, pp. 65–72.

[18] K. Tsarava, K. Moeller, N. Pinkwart, and M. Ninaus, "Training Computational Thinking: Game-Based Unplugged and Plugged-in Activities in Primary School," in *Proceedings of the 11th European Conference on Game-Based Learning EGBL 2017*, 2017, pp. 687–695.

[19] "Digital game Jig Saw Doku." [Online]. Available: <http://www.jigsawdoku.com/>. [Accessed: 15-Mar-2019].

[20] "Digital game Kakuro." [Online]. Available: <https://www.kakuros.com>. [Accessed: 15-Mar-2019].

[21] C. C. Inc., "Code Combat." [Online]. Available: <https://codecombat.com/play>. [Accessed: 15-Mar-2019].

[22] "Initial Conditions." [Online]. Available: <https://reheated.org/games/initial/>. [Accessed: 15-Mar-2019].

[23] "Digital game Thinking Myself." [Online]. Available: <http://games.thinkingmyself.com/>. [Accessed: 15-Mar-2019].

[24] Reheated, "Bebras Challenge," 2017. [Online]. Available: https://challenge.bebaschallenge.org/index.php?action=user_competitions. [Accessed: 15-Mar-2019].

- [25] Code.org, "Digital game Bee - loops." [Online]. Available: <https://studio.code.org/s/course2/stage/8/puzzle/1>. [Accessed: 16-Mar-2019].
- [26] N. Hoić-Božić, M. Holenko Dlab, L. Načinović Prskalo, J. Rugej, and I. Nančovska Šerbec, "Project GLAT-Encouraging algorithmic thinking using didactic games," *Int. J. Multidiscip. Res.*, vol. 4, no. 2, pp. 73–95, 2018.
- [27] N. Hoic-Bozic, M. Holenko Dlab, and V. Mornar, "Recommender System and Web 2.0 Tools to Enhance a Blended Learning Model," *IEEE Trans. Educ.*, vol. 59, no. 1, 2016.
- [28] Google, "Blockly Demo Plane Seat Calculator." [Online]. Available: <https://blockly-demo.appspot.com/static/demos/plane/index.html>. [Accessed: 14-Mar-2019].
- [29] Code.org, "Digital game Code with Anna and Elsa." [Online]. Available: <https://studio.code.org/s/frozen/stage/1/puzzle/1>. [Accessed: 14-Mar-2019].
- [30] LightBot Inc., "Digital game LightBot." [Online]. Available: <http://lightbot.com/flash.html>. [Accessed: 14-Mar-2019].
- [31] Tynker, "Digital game Dragon Dash." [Online]. Available: <https://www.tynker.com/hour-of-code/>. [Accessed: 12-Mar-2019].
- [32] Allcancode Inc., "Digital game Run Marco!" [Online]. Available: <https://runmarco.allcancode.com/>. [Accessed: 14-Mar-2019].
- [33] Code.org, "Digital game Bee - sequence." [Online]. Available: <https://studio.code.org/s/course1/stage/7/puzzle/1>. [Accessed: 14-Mar-2019].
- [34] Code.org, "Digital game Bee - loops." [Online]. Available: <https://studio.code.org/s/course1/stage/14/puzzle/1>. [Accessed: 14-Mar-2019].
- [35] Code.org, "Digital game Bee - conditions." [Online]. Available: <https://studio.code.org/s/course3/stage/7/puzzle/1>. [Accessed: 14-Mar-2019].
- [36] J. Mezak and P. Pejić Papak, "Learning scenarios and encouraging algorithmic thinking," in *41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2018, pp. 836–841.
- [37] GLAT project, "Learning scenarios," 2018. [Online]. Available: https://glat.uniri.hr/?page_id=2371. [Accessed: 19-Mar-2019].