# Serious Games for Learning Programming Concepts

**Ivona Frankovic[1], Natasa Hoic-Bozic[2], Lucia Nacinovic Prskalo[3]**

## Abstract

*Serious games are specially designed computer games in which education is the primary goal, rather than entertainment. They are interactive competitive lessons with defined learning outcomes and their importance in contemporary educational practice is increasing. The implementation of serious games in teaching has a potential to facilitate the learning process in terms of increasing students' interest in learning to ensure better understanding of learning materials and application of acquired knowledge. Serious games can be used to stimulate programming learning. Besides learning coding skills, programming supports the development of computational thinking, which is widely applicable and useful, not just in computer science, but also in everyday life. Programming is recognized as a crucial skill, even a new literacy, for all children, and there is the emerging need to introduce programming concepts into primary schools from the first to the fourth grade. Computational thinking represents an increasingly important focus in informatics (computer science), and ways of incorporating it into the school curricula are being explored. Serious games offer an exciting opportunity for learners to engage in computational thinking. This paper presents an overview of the game genres suitable for better understanding of certain programming concepts. Games and tasks that were taken into consideration trigger the development of computational thinking skills by incorporating components such as abstraction, decomposition, evaluation, and generalization. The examples of serious games are given and classified according to the main programming concepts. This classification can be a starting point for selecting and designing adequate serious games to support the effective learning of programming concepts in the classroom.*

**Keywords**: *serious games, computational thinking, primary education, game-based learning;*

## 1. Introduction

Didactic games can be defined as interactive, competitive lessons with defined learning outcomes that enable students to have fun during acquiring knowledge. Their goal is not merely fun but they contain an educational component as well [1][2].

As didactic video games encourage student motivation and facilitate the learning of complex materials, they are increasingly being used in informatics (computer science) teaching [3]. The main goal of the games in informatics teaching is to introduce students to the world of complex problems, and they are particularly important for the development of computational thinking (CT) and programming learning [4].

The question that arises is to what extent the CT development is associated with programming and whether it should be developed before the programming learning itself, or it is sufficient to introduce it simultaneously at the same time when programming. According to [5] CT can be separated from programming, and should be taught before programming teaching starts.

In order to successfully use games for CT and programming, it is necessary to explore which games and logical tasks are appropriate for the development of particular concepts, i.e. a framework for programming learning for younger students using game based learning (GBL) needs to be developed. As a first step of this research, this paper presents work-in-progress with the aim to identify serious games that are suitable for understanding basic concepts of computational thinking.

## 2. Computational thinking

The best way to teach students coding is to teach the key coding skills first. It is necessary to start with the essential elements for the key building blocks of computer literacy. Many coding skills are not just important for programming, they are critical skills for any career. It is more important to help children gain the thinking skills than to actually write codes. It is widely accepted that students need to demonstrate an understanding of the patterns evident in programming rather than focusing only on syntax and semantics of programming [6][7].

[1] Department of Informatics, University of Rijeka, Croatia
[2] Department of Informatics, University of Rijeka, Croatia
[3] Department of Informatics, University of Rijeka, Croatia

Computational thinking is a general analytic approach to problem-solving, designing systems and understanding human behaviour concerned with conceptualizing, developing abstractions and designing systems that overlap with logical thinking and requires concept fundamental to computing [8][9]. Some studies support the idea that everyone should think computational and that it is crucial for children to develop CT skills before formal programming learning [10] [11], while others claim it is questionable what skills and abilities develop CT and how it should be integrated into education [12].
The most suitable game types for problem solving are puzzles, simulation games, strategy games, adventure games, artificial life and management games [13]. The need for the introduction of coding and the development of computational thinking in primary schools has already been recognized and competitions such as Hackathon on Coding, Bebras are one way of rewarding excellence in that area.

## 3. Games and logical tasks for learning computational thinking

The aim of the Bebras challenge [14] is to promote students' interest in informatics learning from the beginning of their education by solving short tasks, and deepen computational thinking. The most important components for developing computational thinking are interesting logical tasks or puzzles.
In the papers [15] and [6], five key computational thinking skills are suggested:

- **Abstraction** makes problems easier to think about by spotting key elements in certain problem/task and removing unnecessary details without losing any important information.
- **Decomposition** is the way of thinking about problems in terms of their component parts that can be understood, solved, developed and evaluated separately.
- **Algorithmic thinking** is the ability to think in terms of sequence and rules as a way of solving problems and it needs to be applied when steps of problems repeat in similar sequence. It involves creating and executing an algorithm.
- **Evaluation** is the process of ensuring that the obtained solution is suitable for the given purpose.
- **Generalization** is a way of quickly solving new problems based on the previous solutions, and building on the earlier experience. It is related to identifying patterns, similarities and connections.

In [15] a two-dimensional categorization system, which incorporates CT skills and informatics concepts is introduced. A task should be assigned to only one informatics concept and up to three CT skills.
Brennan & Resnick [16] define a framework for CT which is composed of three key dimensions: computational thinking concepts, computational thinking practices, computational thinking. They have also defined the following concepts that can apply to programming and non-programming context: sequence, loops, events, conditionals, operators, data and parallelism.
The most interesting dimensions at the beginning of our research are computational thinking concepts, which are common in programming approach. As a first step, we wanted to show the connections of the CT concepts with the particular games and tasks (Table 1). The examples of the tasks are taken from the Bebras contest [14], while the examples of the games are taken from the following sites: *Education, Blockly games* and similar. It is planned to create such types of games in the Croatian language.

Table 1. Connections of the CT concepts with the particular games and tasks

| **Computational thinking concepts** | **Game genre** | **Game example** | **Task example** | **Description** |
|---|---|---|---|---|
| Decomposition | Sudoku | Picture sudoku<br>Kakuro for kids | Animation<br>Beach Flags<br>Broken Window<br>Beaver Code | Solving the sudoku encourages the discerning of the exact rules of character scheduling. It is important to understand the rule and know how to apply it in different situations – from the easier ones when only one character is missing to the ones that are more difficult when more characters are missing. |
| Abstraction | Doodle dots<br>Tetris<br>Jigsaw puzzle<br>Tangram games | Patchwork<br>Doodle dots | Mushrooms<br>Bracelet<br>Walnut Animals<br>Geocaching | By solving these tasks and games, children learn to use and interpret symbols or representations in order to think through and solve a variety of problems. |
| Algorithmic Thinking | Puzzle games<br>Maze games | Blockly Games: Bird<br>Run Marco<br>Blockly Games: Maze | Setting the Table<br>Crane Operating<br>Fair Share<br>Candy jar<br>Cross Country<br>Beaver Code | To solve these tasks, it is important to set a proper sequence of steps/commands. If they are not set in the proper order, the task cannot be solved. By identifying the patterns in the tasks, we can make predictions, create rules and solve problems that are more general. |
| Evaluation | Memory game<br>Jigsaw puzzle | Blockly Games: Puzzle<br>Matching games | Dream Dress<br>Geocaching<br>Cross Country | The tasks involve statements (conditions/requirements) that must be evaluated (determined to be true or false) for a set of objects. Conditions and their evaluation are important because the decisions are made based on them. |
| Generalization | Pattern games | Pattern games | Beaver Code<br>Birthday Balloons<br>Animation | Resolving a sequence of items enables students to practice the ability to recognize the rules of changing the elements in a row and to use these rules to predict the next step. These tasks help us to realize that certain information can assist in the prediction of what follows. |

## 3. Conclusion and future work

This paper presents a work-in-progress research with the aim of creating a framework for programming learning in the younger age of students. The basis of the framework will be a two-dimensional categorization consisting of programming concepts and computational thinking concepts. CT concepts and concepts of programming will be linked to the games and logical tasks that help in their learning.

So far, the construction of the classification for CT has begun and the first version is presented in this paper. The following steps include extending the classification with the games for programming concepts, game collection, game and logical tasks development in the Croatian language, as well as the implementation of experiments in primary schools, where students from first to fourth grade will test the proposed framework in programming learning.

## 4. 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).

## References

[1] Blumberg, F. C., Almonte, D. E., Anthony, J. S., and Hashimoto, N., “Serious Games: What Are They? What Do They Do? Why Should We Play Them?,” The Oxford Handbook of Media Psychology, 2013, pp. 334–351.
[2] Bourgonjon, J., De Grove, F., De Smet, C., Van Looy, J., Soetaert, R., and Valcke, M., “Acceptance of game-based learning by secondary school teachers,” Computers and Education, vol. 67, 2013, pp. 21–35.
[3] Li, M.-C. and Tsai, C.-C., “Game-Based Learning in Science Education: A Review of Relevant Research,” Journal of Science Education and Technology, vol. 22, no. 6, 2013, pp. 877–898.
[4] Bond, C., “Journal of Computer Assisted Learning,” Anzmac, vol. 3, no. 1, 2010, pp. 1–9.
[5] Lu, J. J. and Fletcher, G. H. L., “Thinking About Computational Thinking Categories and Subject Descriptors,” pp. 6–10.
[6] Csizmadia, A., Curzon, P., Humphreys, S., Ng, T., Selby, C., and Woollard, J., “Computational thinking: A guide for teachers,” 2015, pp. 1–18.
[7] Kazimoglu, C., Kiernan, M., Bacon, L., and MacKinnon, L., “Understanding Computational Thinking before Programming,” International Journal of Game-Based Learning, vol. 1, no. 3, 2011, pp. 30–52.
[8] Wing, J. M., “Computational thinking,” vol. 49, no. 3, 2006, pp. 33–35.
[9] Wing, J. M., “Computational thinking and thinking about computing,” Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 366, no. 1881, 2008, pp. 3717–3725.
[10] Perković, L., Settle, A., Hwang, S., and Jones, J., “A framework for computational thinking across the curriculum,” Proceedings of the fifteenth annual conference on Innovation and technology in computer science education - ITiCSE ’10, 2010, p. 123.
[11] Jake, Q. A. and Sherrell, L. B., “Why computational thinking should be integrated into the curriculum.” 2010, pp. 66–70.
[12] Lee, I. … Werner, L., “Computational thinking for youth in practice,” ACM Inroads, vol. 2, no. 1, 2011, p. 32.
[13] Vahldick, A., Mendes, A. J., and Marcelino, M. J., “A review of games designed to improve introductory computer programming competencies,” Proceedings - Frontiers in Education Conference, FIE, vol. 2015–Febru, no. February, 2015.
[14] “Bebras.” [Online]. Available: http://www.bebras.org/.
[15] DAGIENĖ, V., SENTANCE, S., and STUPURIENĖ, G., “Developing a Two-Dimensional Categorization System for Educational Tasks in Informatics,” Informatica, vol. 28, no. 1, 2017, pp. 23–44.
[16] Brennan, K. and Resnick, M., “New frameworks for studying and assessing the development of computational thinking,” annual American Educational Research Association meeting, Vancouver, BC, Canada, 2012, pp. 1–25.
[17] https://www.education.com
[18] https://blockly-games.appspot.com/