DIVERSE CATEGORIES OF PROGRAMMING LEARNING ACTIVITIES COULD BE PERFORMED WITHIN STORYTELLING ALICE

Maria Kordaki, Panagiotis Psomos

University of the Aegean, Department of Cultural Technology & Communication (GREECE)
m.kordaki@aegean.gr, panagiotis.psomos@aegean.gr

Abstract

A set of eleven categories of learning activities is presented in this study which could be performed by the students using the tools of the well-known educational software Storytelling Alice that is dedicated for the learning of programming by novices. Specifically, eleven categories of learning activities that could be performed within Storytelling Alice were formed, namely: (i) Free creative activities, (ii) Creating a specific story, (iii) Multiple solution tasks, (iv) Experimentation within working Storytelling Alice projects, (v) Modification of working Storytelling Alice projects, (vi) Working on a complete Storytelling Alice story and a correct but incomplete part of its code, (vii) Working on a complete Storytelling Alice story and a mixed form of its code, (viii) Working with a complete Storytelling Alice story and an incorrect part of its code, (ix) Working with the complete code of a Storytelling Alice story and predicting its output, (x) Black-box activities, and (xi) Collaborative learning activities. The aforementioned categories of learning activities can be used in the computer science classroom to support novices in learning programming within Storytelling Alice.

Keywords: Programming, Storytelling Alice, educational software.

1 INTRODUCTION

Programming is a complex task that can be divided into four steps: a) understanding of the problem at hand, b) definition of a solution to that problem - initially in any form, such as text-based or math-based and then, in a computer compatible form such as pseudo-code and flow-chart, c) translation of the solution into a selected programming language, and d) testing and debugging of the resulting program. It is acknowledged, that students encounter serious difficulties in performing all the steps mentioned above [1][2][3][4][5][6][7][8]. Many students may lack the necessary domain background to face the problems at hand, as programming problems come from a wide range of problem domains [6]. Some other students, after finding a solution, usually tend to omit the “algorithmic solution” and confront the problem directly [1]. Most importantly, novices are rarely aware of the problems that can be solved by a computer and the benefits to be had from using programming. In fact, students often begin programming with a “big handicap”: they do not know how a computer works [8] and try to explain its functionality by using anthropomorphic expressions [9]. This lack of knowledge results in many other “deep misconceptions” as well as “surface errors” that appear as programming bugs [2]. In addition, novices –but sometimes also Computer Science students - seem to lack the ability to combine the individual statements and constructs of a programming language into valid programs [6][8]. Furthermore, some students know the syntactical rules of specific statements and constructs of a specific programming language but do not know how to combine these features into correct programs [6][8]. Novices often have difficulties to identify primitive programming constructs such as conditionals, recursions and loops as well as entities such as variables, counters and conditions - in their initial solution; nor can they express them with computer-based structures [8][10] In fact, there is a big gap between the informal solutions they give and the more formal computer-oriented solution that is required [6] and the bigger the gap, the more difficult it is for the student to surpass it [5]. Students also have semantic and syntactic misconceptions of almost every basic structure and concept of a programming language [4][5]. There are also important difficulties in understanding the way input and output commands function and, according to Du Boulay [2], students find it difficult to manipulate arrays. In addition, not only do inexperienced users have difficulties in using a programming language but, in contrast to those with experience, they also find it difficult to comprehend and use the online help provided by the environment [1].

Storytelling Alice is a digital storytelling environment which offers a new programming language that is especially suitable for novices. In this direction, Storytelling Alice does not support the learning of programming through writing long programs in text format using complex structures and following rigid

syntactical rules which are mainly familiar to expert programmers. Storytelling Alice supports constructivist programming learning, by allowing the students to create meaning from their experiences and interactions within authentic contexts, which is vital for constructivism and social views of learning [11]. The promotion of authentic learning activities is a central aspect of constructivist learning that is also viewed as an active, subjective and constructive process. It is also worth noting that the context or activity that frames the construction of knowledge is of equal importance to the learner as the knowledge itself [12]. However, even if there is some progress in solving some learning problems in programming using constructivist learning theories [13] and specific educational software, the problems and difficulties associated with the learning of introductory programming by novices remain to be researched [7].

In this direction, taking into account social and constructivist learning approaches [11][14] an attempt has been made to formulate categories of essential types of learning activities which could be performed within Storytelling Alice. In fact, eleven categories of learning activities that could be performed within Storytelling Alice were formed, namely: (i) Free creative activities, (ii) Creating a specific story, (iii) Multiple solution tasks, (iv) Experimentation within working Storytelling Alice projects, (v) Modification of working Storytelling Alice projects, (v) Working on a complete Storytelling Alice story and a correct but incomplete part of its code, (vii) Working on a complete Storytelling Alice story and a mixed form of its code, (viii) Working with a complete Storytelling Alice story and an incorrect part of its code, (ix) Working with the complete code of a Storytelling Alice story and predicting its output, (x) Black-box activities, and (xi) Collaborative learning activities. All except categories (i), (ii), (iv), (v) and (xi), describe innovative learning activities within the context of Scratch. Computing teachers can use these categories of activities in their attempts to design appropriate every day classroom settings for the learning of programming by novices within Storytelling Alice. This is the contribution of this paper.

In the next section of this paper, a brief description of essential features provided by the Storytelling Alice environment is given, followed by a description of the aforementioned categories of learning activities which could be performed within the program. These categories are also discussed in terms of their possible contribution on students' learning. Finally, the paper ends with a summary and future research plans.

2 FEW WORDS ABOUT STORYTELLING ALICE

Storytelling Alice is an innovative and unprecedented 3D programming environment that aims to transfer pure text programming into programming by creating digital stories [15]. It incorporates digital storytelling in the encoding process, giving thus a visual dimension to the programming process, which makes it easier for novice programmers, especially girls [16]. Through user testing of Storytelling Alice, it was found that users' ability to find and develop story ideas was important in maintaining their engagement with programming in the system [17]. What is more, Storytelling Alice can be used by middle-school students to make games, and this activity has the potential to promote aspects of IT fluency [18]. Storytelling Alice is a similar environment to Alice 2.0. But the environment of Storytelling Alice was designed for use by middle school children, offering a higher level of animations. For example, the Environment Storytelling Alice offers extra characters (such as fairies and characters typical student) and also ready programmed behaviors (e.g. dancing, kissing, talking) targeting middle school students. To enable and encourage users to create animated stories, Storytelling Alice includes: (a) High-level animations that enable users to program social interactions between characters. (b) A story-based tutorial that introduces users to programming through building a story. (c) A gallery of 3D characters and scenery with custom animations designed to spark story ideas.

Storytelling Alice provides a motivating context for learning programming. A study comparing middle school girls' experiences with learning to program in Storytelling Alice and in a version of Alice without storytelling features (Alice 2.0) showed that [16]:

- Users of Storytelling Alice spent 42% more time programming than users of Alice 2.0.
- Users of Storytelling Alice were more than three times as likely to sneak extra time to work on their programs as users of Alice 2.0 (51% of Storytelling Alice users vs. 16% of Alice 2.0 users snuck extra time to program).
- Despite the focus on making programming more fun, users of Storytelling Alice were just as successful at learning basic programming concepts as users of Alice 2.0.

Figure 1: A digital story in the Storytelling Alice environment

3 CATEGORIES OF ESSENTIAL LEARNING ACTIVITIES WHICH COULD BE PERFORMED WITHIN STORYTELLING ALICE

Eleven different learning settings could be formed within Storytelling Alice: First of all, students should be provided with opportunities to freely experiment with the tools of Storytelling Alice to create their own digital stories, animations and interactive arts. Students should also be asked to create a specific story within Storytelling Alice. However, students should also be provided with the chance to perform various learning activities while working with both: complete and incomplete stories within Storytelling Alice. Thus, working within the aforementioned different learning settings, students could perform the following type of programming learning activities within Storytelling Alice:

(i) *Free creative activities*: In this category, students will be able to experiment with the tools of Storytelling Alice to create programs that produce a digital story of their preference. Students' motivation and creative thought can be stimulated from this kind of activity that is not so common in real school practices. Nevertheless, students have to get acquainted with the tools of Storytelling Alice and its possibilities by themselves. Thus, they can use trial and error techniques, perform various explorations on working examples presented in various websites as well as collaborate with other students around the world.

(ii) *Creating a specific story*: Here, students should be asked to create a specific story within Storytelling Alice. Students will be able to immediately see the results on their programming attempts in each step of the creation process and make appropriate corrections.

(iii) *Multiple solution tasks*: In this category, students should be asked to formulate programming solutions to a given task 'in as many ways as possible'. This kind of tasks can challenge the formation of multiple perspectives regarding with a program design by the students [19]. Promoting the development of multiple perspectives during learning settings is a learning approach that provides opportunities for the construction of flexible and meaningful knowledge structures at the same time encouraging the expression of learners' inter- and intra-individual differences [20]. Multiple solution-based learning activities are also acknowledged as essential for the development of problem solving abilities, creativity and advanced mathematical thinking [21]. With the promotion of multiple perspectives, learners also have the chance to become aware that there are multiple approaches to an issue, which is the case in real-life situations. Learners have also the chance to explore each perspective to seek a meaningful resolution to the issue at hand and construct new meanings within the context of their own experiences.

(iv) *Experimentation within working Storytelling Alice stories*: Here, various experimentations van be performed with the specific programming constructs used in the formation of the code of a project (eg. omit specific blocks from the code of the program or change their position, change the value of some

variables, etc) in order to provide explanations and justifications about the role of these constructs so that be able to understand them. This kind of activities could be used as a scaffolding element for students' familiarization with programming constructs and the Storytelling Alice tools so as to avoid some possible frustration from the cognitive load emerging from dealing with problems requiring the use of a 'new' methodology such as algorithmic logic.

(v) *Modification of working Storytelling Alice projects*: In this category, students should be asked to modify the code of a working project in order to produce a slightly different digital story. This is a further step where students are called to actively construct a part of the solution of a problem. By being involved in this kind of activity, students can use the knowledge they acquired during other previous programming activities in order to advance in the modification of the program at hand. At the same time students can feel as sheltered by the context of the already working project in order to appropriately face the challenges of its modification.

(vi) *Working on a complete Storytelling Alice story and a correct but incomplete part of its code*: By being involved in this kind of activity students can reflect on a working digital story as a guide in combination with the incomplete code provided and use their previous knowledge in order to provide the complete code. In order to succeed, Storytelling Alice provides students with the chance to correct their programming attempts through trial and error.

(vii) *Working on a complete Storytelling Alice story and a mixed form of its code*: As far as this category is concerned, students should be provided with a mixed mode of the code of a complete Storytelling Alice digital story and should be asked to make the correct adjustments so that this code produces the given output. This kind of activity can act as a scaffolding element on students' attempts to formulate the correct code as they do not have to invent it from scratch but have all its parts in their disposal and they have to think and experiment till they find the appropriate sequence of commands.

(viii) *Working with a complete Storytelling Alice story and an incorrect part of its code*: In this category, the code provided should address to the students main misconceptions during programming. Then, students should be asked to find the code mistakes and also make appropriate corrections so that this code produces the expected digital story. This kind of activity is a little bit hard for the students to grasp, as it implies that they have to find the specific mistakes included in the given code and also to make corrections so that it produces the given output.

(ix) *Working with the complete code of a Storytelling Alice story and predicting its output*: Due to the fact that making predictions is a high level cognitive activity [22] students are able to use their knowledge in an operative level. This kind of activity implies that students not only know the role of each specific programming construct used in the given code but that they understand what the synthesis of all specific programming constructs in the code really means.

(x) *Black-box activities*: In this category, students can be provided with a complete Storytelling Alice digital story and should be asked to program it. Students can experiment with the output and reflect on their previous knowledge to construct the code. Students are called to use various thinking skills such as reversible thinking, analytical and synthetic thinking, as well as reflection, prediction, hypothesis generation and exploration. The development of this kind of thinking skills go hand by hand with the acquisition of problem-solving and decision-making skills that are potentially essential in both the personal and professional life of a learner.

(xi) *Collaborative learning activities*: With the use of various collaborative patterns (eg. The Jigsaw collaborative pattern; [23]), students' can collaborate in order to perform appropriate programming activities within Storytelling Alice. Students collaboration with their classmates as well as with other students through Internet around the world can be performed through such activities.

4 SUMMARY AND FUTURE RESEARCH PLANS

A set of eleven essential categories of programming learning activities is presented in this paper which could be performed within the well known educational software Storytelling Alice that is dedicated for the novices' learning of programming in a playful and creative way. These categories are: (i) Free creative activities, (ii) Creating a specific story, (iii) Multiple solution tasks, (iv) Experimentation within working Storytelling Alice projects, (v) Modification of working Storytelling Alice projects, (vi) Working on a complete Storytelling Alice story and a correct but incomplete part of its code, (vii) Working on a complete Storytelling Alice story and a mixed form of its code, (viii) Working with a complete Storytelling Alice story and an incorrect part of its code, (ix) Working with the complete code of a

Storytelling Alice story and predicting its output, (x) Black-box activities, and (xi) Collaborative learning activities. All except categories (i), (ii), (iv), (v) and (xi), describe innovative learning activities within the context of Storytelling Alice. The performance of such kind of programming learning activities could encourage students' learning of programming as well as the development of their core thinking skills. The use of this diverse kind of activities in real classrooms to investigate their impact on students' programming thinking is in our future plans.

REFERENCES

[1] Allwood, C. (1986). Novices on the computer: a review of the literature. *International Journal of Man-Machine Studies*, Vol. 25, 633-658.

[2] Du Boulay, B. (1986). Some difficulties in Learning to Program. *Journal of Educational Computing Research*, 2 (1), 57-73.

[3] Soloway, E. & Spohrer, J. C. (1989). *Studying the novice programmer*. Hillside, N.J. Erlbaum.

[4] Lemone, K. A. & Ching, W. (1996). Easing into C: Experiences with RoBOTL. *ACM, SIGCSE Bulletin,* Vol. 28, No. 4, 45-49.

[5] Christiaen, H. (1998). Novice Programming Errors: Misconceptions or Mispresentations? *ACM, SIGCSE Bulletin*, Vol. 20, No. 3, 5-7.

[6] Winslow, L. E. (1996). Programming Pedagogy. *SIGCSE Bulletin,* Vol. 28 (3), 17-22.

[7] Hadjerrouit, S. (2008). Towards a Blended Learning Model for Teaching and Learning

[8] Eckerdal, A. (2009). *Novice Programming Students' Learning of Concepts and Practise*. Dissertation presented at Mathematics and Computer Science, Department of Information Technology, Upsalla University, Sweeden.

[9] Dagdilelis, V., M. Satratzemi and G. Evangelidis (2004). Introducing secondary education to algorithms and programming. *Education and Information Technologies*, 9(2), 159–173.

[10] Lahtinen, E. and Ahoniemi, T. (2009). Kick-Start Activation to Novice Programming - A Visualization-Based Approach. *Electronic Notes in Theoretical Computer Science*, 224 (2009), 125–132.

[11] Jonassen, D. H. (1999). Designing constructivist learning environments. *Instructional design theories and models*, 2, 215-239.

[12] Dabbagh, N. (2005). Pedagogical models for E-Learning: A theory-based design framework. *International Journal of Technology in Teaching and Learning*, 1(1), pp. 25-44.

[13] Ben-Ari, M. (2004). Situated learning in computer science education. *Computer Science Education*, 14(2), 85–100.

[14] Vygotsky, L.S. (1978). *Mind in Society.* Cambridge, MA: Harvard University Press.

[15] Jose, G. (2008). *ALICE, the Programming Language*. Jones and Bartlett Publishers Computer Programming: A Case Study. *Informatics in Education*, 2008, Vol. 7, No. 2, 181–2

[16] Kelleher, C. (2006). Motivating Programming: Using storytelling to make computer programming attractive to middle school girls. PhD Dissertation, Carnegie Mellon University, School of Computer Science Technical Report CMU-CS-06-171

[17] Kelleher, C. (2009) Barriers to Programming Engagement. Advances in Gender and Education, vol. 1, no. 1, pages 5-10.

[18] Werner, L.L., Denner, J., Bliesner, M., and Rex, P. (2009). Can middle-schoolers use Storytelling Alice to make games?: results of a pilot study. In Proceedings of Foundations of Digital Games Conference Series, 207-214.

[19] Kordaki, M. (2009). Beginners' programming attempts to accomplish a multiple-solution based task within a multiple representational computer environment. In (Eds.), Proceedings of World Conference on Educational Multimedia, Hypermedia And Telecommunications (Ed-Media) 2009, June 22-26, 2009, Hawaii, USA. (pp. 3282-3295). Chesapeake, VA: AACE.

[20] Kordaki, M. (2006). Multiple Representation Systems and Students' Inter-Individual Learning

Differences. In P. Kommers & G. Richards (Eds.), Proceedings of *World Conference on Educational Multimedia, Hypermedia and Telecommunications* (ED-MEDIA) 2006 (pp. 2127-2134). Chesapeake, VA: AACE.

[21] Leikin R., Levav-Waynberg A., Gurevich I. & Mednikov L. (2006). Implementation of Multiple Solution Connecting Tasks: Do Students' Attitudes Support Teachers' Reluctance? *Focus on learning problems in mathematics,* 28(1), 1-22.

[22] Marzano, J.R., Brandt, S.R., Hughes, C-S, Jones B-F., Presseisen, Z.B., Rankin, C.S. and Suhor, C. (1988). *Dimensions of Thinking: A Framework for Curriculum and Instruction.* VA: Association for Supervision and Curriculum Development.

[23] Aronson, E., Blaney, N., Sikes, J., Stephan, G., & Snapp, M. (1978). *The JIGSAW classroom*. Beverly Hills, CA: Sage Publications.