Learning programming with serious games

Matej Zapušek[1,*], Jože Rugelj[1]

[1]Univerza v Ljubljani, Pedagoška fakulteta, Kardeljeva ploščad 16, 1000 Ljubljana

Abstract

Students who are learning to program often have difficulties understanding cognitively complex concepts. Teaching programming is mainly focused on the syntax and features of programs, rather than to a deeper understanding of programming constructs and abstract concepts. Computer game stimulates active learning and presentation of learning content in a variety of contexts that are funny and engaging for students. This has a positive impact on the motivation to learn. This paper deals mainly with defining the programming knowledge and common problems with teaching programming, comparing the properties of novice and experts programmers and introducing the semantic method of teaching programming where one would teach only the semantics of programming constructs unbound to specific programming language in an interactive motivating setting of educational computer game. In this paper we discuss the main characteristics of computer games and specific features which makes them useful in the educational setting. As an example of presented method we introduce a game on the presentation of variables in programming. The game is based on visualizations of different types of variables and on the interpretation of the assignment sentence. The game actively encourages interactivity and deeper learning.

Keywords: teaching programming, programming concepts, semantic method, serious computer game.

Received on 26 July 2012; accepted on 07 January 2013; published on 20 March 2013

Copyright © 2013 Zapušek and Rugelj, licensed to ICST. This is an open access article distributed under the terms of the Creative Commons Attribution licence (http://creativecommons.org/licenses/by/3.0/), which permits unlimited use, distribution and reproduction in any medium so long as the original work is properly cited.

doi: 10.4108/trans.gbl.01-06.2013.e6

1. Introduction

Novice programmers often have difficulties in learning programming because concepts are complex, cognitively demanding, and are radically different from what they are accustomed to. Introductory courses are generally regarded as difficult and often have high dropout rates. Learning programming requires algorithmic thinking, problem-solving skills, and it is a long-term process. According to Winslow [1] it takes 10 years for a novice programmer to become an expert. Our goal is to find new didactic approaches to facilitate the learning process and to make it more efficient. Teaching programming traditionally mainly focuses on the syntax and specific features of programming languages, rather than on the deeper understanding of programming constructs and abstract concepts. Our goal is to present semantic aspect of the programming concepts that are not strictly related to a specific programming language. For each concept we want to find an activity that resembles the main idea and present it in the form of educational computer game which has its own positive pedagogic aspects and its not just one of the possible implementations of the proposed method. As an example, we present an educational computer game for teaching the concept of variable which has been developed according to our model.

*Corresponding author. Email: matej.zapusek@pef.uni-lj.si

2. Learning and teaching programming

The main objective of this chapter is to define which competences are composing the programming knowledge domain to be able to adequately address them in the proposed application. We also identify the most common challenges in teaching/learning how to program, so that we can better understand what prevents the novice programmer from more effective knowledge acquisition.

2.1. Programming knowledge

Studies in programming are divided into two main categories: those from the perspective of software engineering and those with psychological/educational perspective [2]. Software engineering perspective focuses mainly on professionals and how they can more effectively develop software projects. The psychological/educational perspective focuses primarily on program comprehension and generation, on mental models, on identifying the properties of novice programmers and on comparing them with experts, knowledge, and skills required for programming. The latter is the focus of our research.

There are several different views on what is involved in learning programming. Du Boulay [4] identifies five overlapping domains that must be mastered by novice programmer and that can be also viewed as a possible source of problems: (1) general orientation, (2) the notional machine, (3) notation, (4) structures, and (5) pragmatics. General orientation is knowledge about what programs are for and what can be done with them, notional machine is a model of the computer as it relates to executing programs, notation domain deals with the syntax and semantics of a particular programming language, structures are schemata or plans with which we help to efficiently solve the problem and pragmatics which are the skills of planning, developing, testing and debugging. Domains are not separated which creates problem because novice has to deal with all of them at once.

Rogalski and Samurcay defined learning programming as acquiring and developing knowledge about programming that is highly complex process. It involves a variety of cognitive activities and mental representations related to: (1) program design, (2) program understanding, (3) modifying, (4) debugging and (5) documenting. Even at level of computer literacy, it requires construction of conceptual knowledge and the structuring of basic operations (such as loops, conditional statements, etc.) into schemas and plans. It requires developing strategies flexible enough to derive benefits from programming aids (programming environment, programming methods) [5].

According to Winslow [1] programming knowledge includes the following steps: (1) a good understanding of the problem or set of requirements, (2) ongoing identification of the programming constructs from the specification, (3) designing an algorithm using pseudo-code, (4) problem solving strategies (top-down, bottom-up, communication, brainstorming, peer learning), (5) algorithmic thinking, (6) strategies for design, (7) conversion from pseudo code into programming language, (8) testing and correcting code until program is bug-free.

There are some other authors that have quite different understanding of programming knowledge. Green sees it as an exploratory process where programs are created opportunistically and incrementally [6] and is not just transcription from internally held representation of the problem. Similarly as Green, Visser and Davies understand programming as incremental problem-solving process where strategy is determined by localized problem-solving episodes and frequent problem re-evaluation [7].

2.2. Problems with teaching programming

Knowledge vs. strategies

Davies [3] made the distinction between programming knowledge (declarative notion, e.g., being able to state how a “for” loop works) and programming strategies (the way knowledge is used and applied, e.g., using a “for” loop appropriately in a program). Introductory courses (and textbooks) are traditionally bound to specific programming language [8][9] and primarily focus on programming knowledge. Student lacking the programming strategies skills has difficulties with combining the programming constructs (conditionals, loops, etc.) into viable solution. At the beginning of the learning process, there is an emphasis on the syntax of programming language and later the focus shifts to semantics [16]. Observations made by Winslow [1] shows that students who understand the syntax and semantics of individual commands cannot translate it into software code.

This can be considered as a major downside of traditional approach and motivation for searching for better solutions.

Novice programmers

If we want to make efficient educational materials that foster the process of learning programming, we have to identify common features of novice programmers and the problems they have. We also have to identify the features of expert programmers in order to find the most optimal learning path that will be used to teach the novice programmers the strategies and knowledge experts have.

Gomes and Mendes [10] specify the reasons why programming is so difficult to teach/learn: (1) it demands a high abstraction level, (2) it needs a good level of both knowledge and practical problem solving techniques, (3) requires a very practical and intensive study, which is quite different from what is required in many other courses (more based in theoretical knowledge, implying extensive reading and some memorization), (4) usually teaching cannot be individualized, due to common classes size, (5) it is mostly dynamic, but usually thought using static materials, (6) teachers’ methodologies many times don’t take into consideration the student’s learning styles. Different students have different learning styles and can have several preferences in the way they learn, (7) Programming languages have a very complex syntax with characteristics defined for professional use and not with pedagogical motivations.

Kölling and Rosenberg note that novices are limited to surface knowledge about the programs and have “line by line” approach. They also don’t spend enough time in planning and testing code. If the code is faulty, they don’t look for another solution but are only trying to fix it with local solutions instead of reformulating the program.

Difficult concepts

Several researches have been trying to identify the concepts that are difficult for novice programmers. Soloway and Spohrer [11] presented the difficulties that students usually have while learning specific programming language constructs in the book "Studying the Novice Programmer". Variable initialization seems to be more difficult to understand than updating or testing variables. Novice programmers also have problems with loops and conditionals as well as with the actions that are “hidden”, like updating variables in “for” loops. Students also often have misconceptions about the functioning of recursive functions. We can ease the understanding with teaching iterative functions first, but it’s not the other way around. Commands that do very similar syntax or have different meanings in different contexts often pose difficulties in writing software code (e.g., "325" and 325 and the word "static" in C). Students don’t have problems with syntax or understanding of basic concepts but with basic design of the program. Lahtinen, Ala-Mutka and Järvinen [14] made a survey among 559 students and 34 professors in which they tried to find the most difficult concepts and how to use the knowledge to improve learning process. According to this research, the most difficult concepts are: recursion, pointers and references, abstract data types, error handling and using the language libraries. Similar research was carried out by Milne and Rowe [15] and their results are comparable to the previous one. They found out that the most difficult concepts are pointers, recursion and other data structures.

Out of all the difficult concepts that are listed above we decided to include the concept of a *variable* in our game. The concept of variable is crucial for understanding. Without knowledge of storing and manipulating with values in computer it is almost impossible to write even the most primitive algorithm or study other programming constructs such as loops or conditionals.

3. Semantic method for teaching programming

In introductory courses we usually find two different approaches in teaching programming. The first one is so called “traditional” approach where introductory presentation of syntax is followed by commented examples, through which we learn programming. The second method is based on graphical programming tools, such as Scratch or Alice 3D, in order to facilitate the transfer of knowledge. Both methods use different approaches to illustrate programming structures. Instead of text input they offer pre-prepared blocks that have to be combined into functioning program following precise predetermined rules. In this way, they can facilitate understanding of the execution of a program and individual programming structures, but the interpretation of concepts is still in the domain of a teacher, independent learning or experimentation.

We propose a new method for teaching programming, i.e. “semantic method”. It is about presenting the main ideas that are behind programming concepts in a new intuitive setting. In that way cognitive apparatus of novice programmer is not overloaded with the syntax of specific programming language, so she can focus on meaning of the concept and not on how to write it to be sound with the compiler. Student is intuitively interacting with teaching material using logic, ideas, and mechanics that we find in programming to solve presented problems. Semantic method forces student to be focused only on understanding the problem and finding the appropriate solution using logic that is behind programming concepts.

In order to prepare the learning material according to the semantic method one should divide each learning unit into specific learning objectives that can be further classified according to their Bloom taxonomic levels. There are several types of activities:

- teaching a new concept or a term,
- using an idea behind a concept to solve a problem,
- optimizing the solutions,
- monitoring the level of achievement of a learning goal,
- tasks which serves as a pre-test for detecting the current understanding of a learning goal in order to offer personalized path through the learning material

From the content of each learning objective, one should obtain the basic idea, concept, or term and transfer its mechanics to the virtual environment. With the process of transforming concepts one should respect the principles of psychology of programming, which allows assuming which concepts will be problematic for students and how they can be made easier for them to understand them.
An example of deconstructing learning unit into learning objectives and mapping them to the activities that resembles the mechanics will be presented in section “Educational computer game - World of variables”.

4. Educational computer game

We present the semantic method in a format of an educational computer game. We see this approach very suitable as it gives us educational benefits that, we believe, can not be achieved otherwise. In order to defend this statement we have to point out the main characteristics of computer games and how can their features be used in an educational setting to foster knowledge transfer.

4.1. Properties of computer games

The review of research in the field of game based learning pointed out that there are no properties that are common to all games and that games belong to the same semantic category only because they bear a “family resemblance” to one another [18],[19]. There are lots of different opinions from various researches in the field about characteristics that make certain activity a game. Johnston suggested that the dynamic visuals, defined goals, applied rules and constant interaction were such features [20]. Thorton claims that the most important aspect of the game is interactivity [21]. Challenge and risk are the main characteristics pointed out by Baranauskas [22]. Malone points out four elements of computer games: fantasy, curiosity, challenge, and control [23]. For Garris, competition, challenge, social interactions, conversion, and fantasy are the most important elements of every game [24]. Prensky states that computer game can be characterised by six key structural elements: rules, goals and objectives, outcomes and feedback, conflict/competition/challenge/opposition, interaction, representation or story. Kirriemuir [25] defines a digital game as one that: provides some visual digital information or substance to one or more players, takes some input from the players, processes the input according to a set of programmed game rules and alters the digital information provided to the players. Authors of the book “Serious games” define game as voluntary activity (a form of freedom) separated from real life (imaginary world that may have or not have relation to real life), absorbs the player’s full attention and is played according to established rules that all players have to follow.

Another aspect of playing the game is intensity of involvement and engagement that games can invoke. Positive experience of being fully engaged in an activity is described as a state of “flow” [32],[23]. Flow represents an optimal state of performance at a task, a sense of enjoyment and control, where an individual’s skills are matched to the challenges faced and derives from activities that are optimally challenging, with clear goals, feedback, high degree of control and where users are absorbed to the extent that they lose a sense of time and self. Prensky [26] summarizes this as: “In the flow state, the challenges presented and your ability to solve them are almost perfectly matched, and you often accomplish things that you didn’t think you could, along with a great deal of pleasure. There can be flow in work, sports, and even learning, such as when concepts become clear and how to solve problems obvious.”

Malone [23] characterised conditions that likely induce state of flow. He claims that: (a) activity should be structured so that player can increase or decrease the level of challenges faced in order to match exactly personal skills with the requirements for action, (b) it should be easy to isolate the activity, at least at the perceptual level, from other stimuli, external or internal, which might interfere with involvement in it, (c) there should be clear criteria for performance; a player should be able to evaluate how well or how poorly (s)he is doing at any time, (d) the activity should provide concrete feedback to the player, so that she can tell how well she is meeting the criteria of performance, (e) the activity ought to have a broad range of challenges, and possibly several qualitatively different ranges of challenge, so that the player may obtain increasingly complex information about different aspects of her/himself.

Killi proposed a model that deals with state of flow in educational computer games. The main purpose of the model is to link gameplay with experiential learning in order to facilitate flow experience. The model describes learning as a cyclic process through direct experience in the game world. The model stresses that activity that is necessary for learning is not merely cognitive but also behavioural. Thus, learning is defined as a construction of cognitive structures through action or practice in the game world. The model does not consider the role of social presence in educational games. However, the social presence may be supported by equipping a game world with tools that enable communication and interaction between players. If the game world supports collaboration, a role of the teacher is to set up a climate that facilitates collaboration. Further, the teacher selects the content and challenges of the game. These challenges based on educational objectives form the heart of the model. At the core of the task is sustaining the motivation and engagement of the player by pumping appropriate challenges to him or her. The experiential gaming model emphasizes cognitive presence by stressing the importance of reflective observation and schemata construction. In the game, player can test solutions produced and observe the outcomes of actions performed which may lead to the construction of schemata and enable the discovery of new and better solutions to the perceived challenges or problems [33].

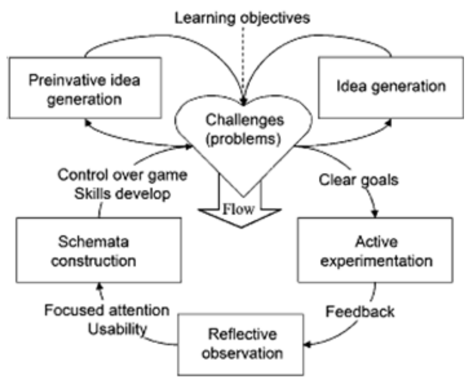

Figure 1. Experimental gaming model

Games can also have positive impact on student's motivation. Motivated learner is enthusiastic, focused, engaged, interested, tries hard, persists over time, is self-determined and driven by its own volition which results in enhanced learning and in accomplishing instructional objectives. Self-determined learner's behaviour can stem from both intrinsic motivation (i.e., the learner engages in an activity because it is interesting or enjoyable) and from extrinsic motivation that they termed identified regulation (i.e., the learner engages in the activity because he or she desires the outcome and values it as important) [34]. Computer games motivate via fun [35], instant visual feedback [26], challenge, curiosity and fantasy [23], active participation, intrinsic and prompt feedback, challenging but achievable goals and mix of uncertainty and open-endedness [36].

4.2. Using computer games for educational purposes

There are several reasons that draw educator attention to games. In formal education we experience a shift from traditional didactic model, which is focused on instruction, to learner-centred model that emphasizes the active learner's role. We also changed the view of learning goals from lower taxonomic levels, like just recalling information, to higher levels, such as finding and using of information in a new settings.

Game based learning was defined by Prensky [26] and Gee [27] as a process of learning with the use of digital games. Games can provide motivation for learning, thus increasing the chance that the desired learning outcomes will be achieved. Learning is defined as the acquisition of knowledge or skills through experience or practice, and what better way to learn than through a game [28]. Almost all studies about game based learning show that students are highly motivated when learning materials are presented in a computer game format and that this has positive effects on the acceleration of a learning process. Students need motivation to focus on what needs to be learned but for any quality learning to occur this is not enough. Comparing learning outcomes of students who learned with computer games and those learning with another type of learning materials shows that there is no significant difference between them. This is usually because of inappropriate game design. Games can be very appealing to students but if they entertain and not teach, the use of games in education does not make much sense. So we have to find out what are the elements that make computer game an educational computer game.

Gross [29] claims that digital games for educational purposes must have well defined learning goals and have to promote development of important strategies and skills to increase cognitive and intellectual abilities of learners.

According to Malone [23] and Garris [24], the elements contributing to educational values of digital games are sensual stimuli (visual and audio representations of learning material), fantasy (context presented in imaginary setting), challenge (demanding or stimulating situation) and curiosity (desire to know or learn). These elements must be incorporated on an integrated platform, to structure objectives and rules, a context of meaningful learning, an appealing story, immediate feedback, a high level of interactivity, challenge and competition, random elements of surprise, and rich environments for learning [23][24].

A game can be instantiated for learning as it involves mental (and sometimes physical) stimulation and develops practical skills – it forces the player to decide, to choose, to define priorities, to solve problems, etc. Immediate reward (and feedback) is a major motivational factor, whether it is translated as game entities (more life power, access to new levels, etc.) or as neurological impulses (happiness, feeling of achievement, etc.). Games can be social environments, sometimes involving large distributed communities. They imply self-learning abilities (players are often required to seek out information to master the game itself), allow transfer of learning from other realities, and are inherently experiential with the engagement of multiple senses [30].

Van Eck [31] argues that games and play can be effective learning environments not because they are fun but because they are immersive, require player to make frequent important decisions, have clever goals, adapt to each player individually and involve social network.

If we consider a model of game based learning by Garris [24], the main characteristic of educational game is that instructional content is blurred within game characteristics. Students are playing the game and having fun, forgetting about the "learning" part of the experience even though they are constantly presented with new concepts which they have to adapt in order to be successful in game. We should foster motivation with game design promoting repeating the cycles within game context. Player is expected to elicit desirable behaviours based on emotional and cognitive reactions that result from interaction with and feedback from gameplay.

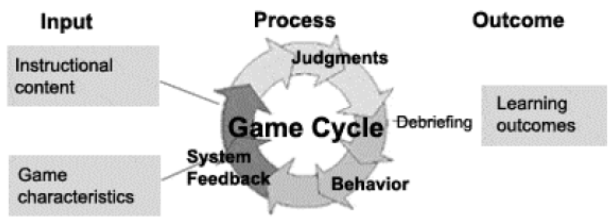

Figure 2. Garris – Game Cycle

Gee [27] argues that features of video games with high learning potential fall into two categories: 'non-game' features, which may also appear in non-game contexts, and 'game' features, which relate more to the 'gameness' of games. Despite this distinction, it should not be assumed that the 'non-game' features would work as well for learning if they were detached from the 'game'.

'Non-game' features of games with high learning potential are:

- Empathy for complex systems – to look at complex system from the inside in order to understand how its variables are interacting.
- Simulations of experience and preparations for action – in video games, players see the virtual world in terms of how it affords the sorts of actions they need to take to accomplish the goal of winning.
- Distributed intelligence via the creation of smart tool - good video games distributes intelligence between a real-world person and artificially intelligent virtual characters in order to represent macro and micro view of the situation.
- Cross-functional teamwork – Good video games may be able to teach collaboration and cross-functional teamwork for institutions like schools and workplaces. In multiplayer games like World of WarCraft groups are composed of different character types, such as hunter, warrior or priest, who each play the game in a different way. Players interact with each other not in terms of their real-world characteristics but through their functional gaming identities. They may also choose to use their real-world race, class, culture, and gender as strategic resources but they are not forced into pre-set racial, gender, cultural or class categories.
- Situated meaning – Dialogue and experience are essential for people to be able to relate words to actual actions, functions, and problem solving. Since video games are simulations of experience, they can situate language in specific contexts.
- Open-endedness: melding the personal and the social – In good open-ended games, players construct their own goals, which are based on their own desires, styles and backgrounds. Combination of personal and in game goals produces a state of high motivation.

Features of a 'good game' that relates to effective learning:

- Motivation - Video games are profoundly motivating for players, and it is important to understand the sources of this motivation if it is to provide a foundation for learning.
- The role of failure – the price of failure is lowered and is often seen as a way to learn the underlying pattern. These features of failure in games allow players to take risks that might be too costly.
- Competition and collaboration – Many gamers, including young ones, enjoy competition with other players in games but may not see competition as pleasurable and motivating in school. Competition in video games is seen by gamers as social, as much about gaming as winning and losing.
- The design of games that relates to: Interactivity – player doesn't just passively consume knowledge but has control over content; Customisation – based on learning styles and providing multiple routes to success; Strong identities - Good games offer players identities that trigger a deep investment on the part of the player and which are clearly associated with the functions, skills and goals one has to carry out in the virtual world; Well-sequenced problems - In connectionist approaches to learning, it is argued that sequencing is crucial for effective learning in complex domains; A pleasant level of frustration - adjusting challenges in such a way that a range of players can experience the game as challenging but do-able; A cycle of expertise - repeated cycles of extended practice and tests of mastery; 'Deep' and 'fair' – game must be challenging but set up in a way that leads to success. Gameplay elements should be initially simple and easy to learn and become more complex the more the player comes to master them.

5. World of variables

We implemented a prototype of the game using Flash with Actionscript 3. In this way we made game easily accessible to interested audience. The game can be found on a web address http://hrast.pef.uni-lj.si/~svet_spremenljivk. Learning objectives of the game are as follows.
Student:

- understands variable as value that is stored in memory of the computer,
- knows that variables have their labels,
- knows there are different types of variables (integer, real, boolean, char…),
- knows that different types of variables are not compatible,
- understands that different types of variables requires different amount of memory to be stored,
- understands the meaning of assigning the value to the variable,
- knows pseudo-code for assignment sentence
- knows the order in which the new value is calculated and assigned to the variable,
- knows the value of the variable after assignment,
- can predict the value of the variable after several assignments.

Game scenario

The game takes place on a planet far away in the universe where inhabitants have a problem making an order from the chaotic mess that dominates on the planet. They also need help with transporting some goods to the other planets. The goods that are transported in the game have similar properties as variables in programming. In this way students can learn the concept of a variable in an intuitive and motivating setting of educational computer game. The game consists of four parts.

Cleaning the mess

Learning goal: declaration of a variable (reservation of memory locations)
Activity: Player has to choose appropriate container (declaration), sticker (label), and at the end the content (assigning the value).

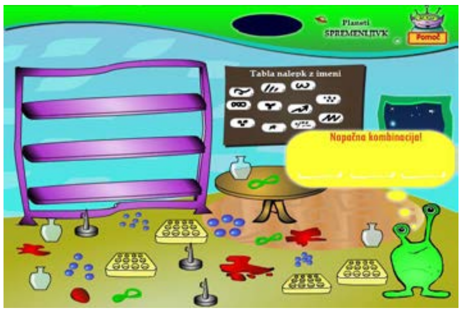

Figure 3. First game – Cleaning the mess

Transporters

Learning goal: understanding there are several types of variables.
Activity: There are transporters that can carry specific goods that resemble different types of variables. Player is placing the goods on appropriate space ship.

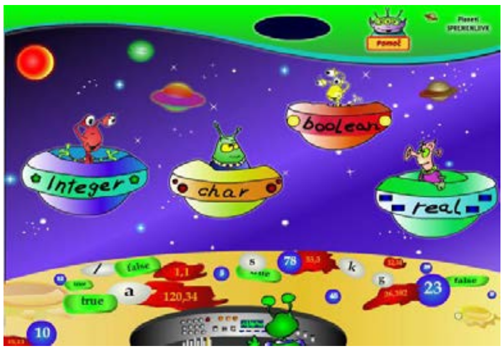

Figure 4. Second game – Transporters

Acquiring the driving licence

Learning goal: initialization – storing a value in memory location (a:=5;)
Activity: The examiner is asking questions about initialization of variables. Player has to choose between four possible answers. Questions are getting more difficult as exam progresses.

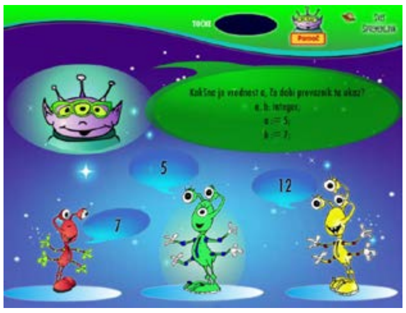

Figure 4. Third game – Acquiring the driving license

Acquisition of the goods

Learning goal: implicit value assignment to the variable (a:=b;)
Activity: Transporter has delivered some goods and player has to help him to store them. Goods have to be placed to the right places so that the robot helper can rearrange them (assigment).

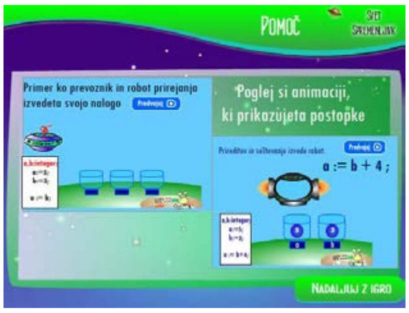
POMOČ
Svet SPREMENLJIVK
Primer ko prevoznik in robot prirejanja izvedeta svojo nalogo
Predvajaj
Poglej si animaciji, ki prikazujeta postopke
Predvajaj
a := b + 4 ;
NADALJUJ Z IGRO

Figure 4. Fourth game – Acquisition of the goods

6. Conclusion

The research findings in the field of psychology of programming show that students who learn to program often have problem to understand cognitively complex concepts. Computer game stimulates active learning and presentation of learning content in different contexts, which are funny and engaging for students. This can have a positive impact on motivation to learn. We can use the game format to teach programming in a way that we carefully examine the mechanics behind a certain programming concept and then we find an appropriate in game activity that resembles its logic. The presented game was designed according to this approach and is based on visualizations of different variable types and on the explanation of the assignment sentence. The game actively promotes interactivity and deeper learning.

Game platform will be used in the future for presenting other topics, especially the concepts that were recognized as difficult to understand. In our further research we will explore the possibilities for efficient modelling of students and programming knowledge.

References

[1] Winslow, L.E. (1996). Programming pedagogy – A psychological overview. SIGCSE Bulletin, Vol. 28, pp 17-22.

[2] Robins, A., Rountree J., Rountree N. (2003). Learning and Teaching Programming: A Review and Discussion. Computer Science Education, Vol. 13, No. 2, pp 137-172.

[3] Davies, S.P. (1993). Models and theories of programming strategy. International Journal of Man-Machine Studies, Vol. 39, pp. 313-320.

[4] du Boulay, B. (1989). Some difficulties of learning to program. In E. Soloway & J.C. Spohrer (Eds.), pp. 283-299.

[5] Rogalski, J. & Samurcay, R. (1990). Acquisition of programming knowledge and skills. In J.M. Hoc, T.R.G. Green, R. Samurcay, & D.J. Gillmore (Eds.), Psychology of programming, pp. 157-174. London: Academic Press.

[6] Green, T.R.G. (1990). Programming languages as information structures. In J.M. Hoc, T.R.G. Green, R. Samurcay, & D.J. Gillmore (Eds.), Psychology of programming, pp. 117-137. London: Academic Press.

[7] Davies, S.P. (1993). Models and theories of programming strategy. International Journal of Man-Machine Studies, Vol. 39, pp. 237-267.

[8] Robins, A., Rountree, J., Rountree, N. (2003). Learning and Teaching Programming: A review and discussion. Computer Science Education, Vol. 13 (2), pp. 137-172.

[9] Jenkins, T. (2002). On the Difficulty of Learning to Program. Acquired at 10.7.2012 from: http://www.ics.ltsn.ac.uk//pub/conf2002/jenkins.html

[10] Gomes, A., Mendes, A. J. (2007). An environment to Improve Programming Education. CompSysTech '07 Proceedings of the 2007 International Conference on Computer Systems and Technologies, Vol. 88. New York: ACM.

[11] Soloway, E. & Spohrer, J. (1989). Studying the Novice Programmer, Lawrence Erlbaum Associates, Hillsdale, New Jersey. 497 p.

[12] Kessler, C. & Anderson, J. (1989). Learning flow of control: recursive and iterative procedures. In Soloway & Spohrer: Studying the Novice Programmer, pp. 229-260.

[13] Rist, R. (1996). Teaching Eiffel as a first language. Journal of Object-Oriented Programming, 9, pp. 30-41.

[14] Lahtinen, E., Ala-Mutka, K., Jarvinen, H.M. (2005). A study of the difficulties of novice programmers. ITiCSE '05 Proceedings of the 10th annual SIGCSE conference on Innovation and technology in Computer Science Education, pp. 14-18. New York: ACM.

[15] Milne, I., Rowe, G. (2002). Difficulties in Learning and teaching programming – Views of Students and Tutors. Education and Information Technologies, Vol. 7 (1), pp. 55-66.

[16] Booth, S. B. Sc. (1992). Learning to program, a phenomenographic perspective. Gotenborg: Studies in Educational Sciences 89.

[17] Thomsen, B. (2008). Using On – LineTutorials in Introductory IT courses. V.J. Bennedsen, M.E. Caspersen, M. Kolling (Ur.), Reflections on the Teaching of Programming. Methods and Implementations, pp. 68. Berlin: Springer.

[18] Wittgenstein, L. (1953). Philosophical investigations. New York: Macmillan.

[19] Wittgenstein, L. (1958). The blue and brown books. New York: Harper & Row.

[20] Johnston,R.T., de Felix, W. (1993). *Learning from video games*. Computer in the Schools, 9, 199-233.

[21] Thornton, G.C. & Cleveland, J.N. (1990). *Developing managerial talent through simulation*. American Psychologist, 45, 190-199.

[22] Baranauskas, M., Neto, N., & Borges, M. (1999). *Learning at work through a multi-user synchronous simulation game*. Proceeding of the PEG'99 Conference, Exeter, UK(137-144). Exeter, UK: University of Exeter.

[23] Malone, T.W. (1981). Toward a theory of intrinsically motivating instruction. Cognitive Science 4.

[24] Garris, R., Ahlers, R., & Driskell, J. E. (2002). Games, motivation, and learning: A research and practice model. Simulation & Gaming, 33(4), 441-467.

[25] Kirriemuir, J., McFarlane A. (2004). Literature review in games and learning.

[26] Prensky, M. (2001). *Digital Game-based Learning*. McGraw-Hill

[27] Gee, J. P. (2003). What video games have to teach us about learning and literacy. New York: Palgrave Macmillan.

[28] Pivec M., & Kearney, P. (2007). *Games for Learning and Learning from Games*. Informatica 31. Pp 419-423

[29] Gross, B. (2003). *The impact of videogames in education*. First Monday, v. 8, n. 7, jul. 2003.

[30] Baptista R., Vaz de Carvalho, C. (2010). *Role Play Gaming and Learning*. Learning Technology, volume: 12, Issue: 1.

[31] Van Eck R. (2006), Digital Game-Based Learning: It's Not Just the Digital Natives Who Are Restless, *EDUCAUSE Review*, Vol. 41, No. 2.

[32] Csikszentmihalyi, M. (1990). Flow: The psychology of optimal performance. New York: Cambridge University Press.

[33] Kiili, K.: Digital Games-based Learning: Towards an Experiential Gaming Model. The Internet and Higher Education, Vol.8, Issue 3 (2005) 13-24.

[34] Deci, E. L., & Ryan, R. M. (1985). Intrinsic motivation and self-determination in human behavior. New York: Plenum.

[35] Bisson C, Luckner J (1996). Fun in learning: the pedagogical role of fun in adventure education. *Journal of Experimental Education*, 19(2), 108–112.

[36] Becta (2002). What is the educational value of computer and video games? ICT Advice Sheet.