INITIATING IN PROGRAMMING VIA EDUCATIONAL GAME

JAN OLESEN
Aalborg University Copenhagen, Denmark

RADOSLAV STOJIC
Belgrade Metropolitan University, Radoslav.stojic@fit.edu.rs

OLGA TIMCENKO
Aalborg University Copenhagen, Denmark, ot@create.aau.dk

Abstract: Although teaching how to program a computer is not a new task, there is still a need to make this process more efficient. This paper analyzes how to initiate novices in the secrets of programming by putting the learning objectives in the game context. The idea is to exploit students' motivation to explore the game world by populating it with some programming entities. In order to progress through the game, students need to show some knowledge and understanding of the presented programming information. The game is tested on two on-line forums with programming and gaming audience. Test results show encouraging data, as this approach has gained some interest and attention.

Keywords: *E-Learning, Gamification of learning, Motivation and e-Learning, Computer games, Educational games*

1. INTRODUCTION

Data shows that dropout rate because of introduction programming courses is very high among university students, even on Computer science programs. Taking into account that nowadays programming is taught at many different educational programs, ranging from engineering to media studies, this problem becomes even more important.

What most researchers mention as the biggest challenge in teaching programming is the lack of motivation among students. What seems to be a pattern among the researches is the idea of turning exercises into something atractive that would increase the motivation. This new way could be the use of computer games as teaching media.This is often tried in a form of "programming put in a game context". Garris et. al. finds that "*games seem to be effective in enhancing motivation and increasing student interest in subject matter...*"[1], which seems to be exactly what is needed. References [2] and [3] report that this aprouch leads to some success.

When talking about games with an educational purpose the term *serious games* is often used. However, when looking for research regarding the combination of serious games and introductory programming or something equivalent there seems to be little to no results.

This paper investigates a combination of a serious game and introductory programming as a way to teach about programming and the problem considered is:
"To what extent can a serious puzzle game introducing C++ motivate the player?" The paper is organized as follows: The first chapter briefly discusses serious games and elements which differ serious games from ordinary games. Several existing related educational games are also presented, together with analysis of their impact. Analysis of the possible target audience follows, together with rationale why more research in this area is needed.

In the next section, design requirements for the new game are analyzed and argued for. For an educational game, it is important to be able to measure whether the players have in fact learned anything - so test procedure will be presented and analyzed in detail. Testing results will be presented as well.

2. SERIOUS GAME DESIGN GUIDELINES

There is no precise definition of serious games, though a general consensus seems to be "*games for purposes other than entertainment*". In practice "other purpose" often reduces to learning. According to Zyda [4], in order for a computer game to become a serious game, addition of pedagogy which is referred to as "*activities that educate or instruct, thereby imparting knowledge or skill*", is required.

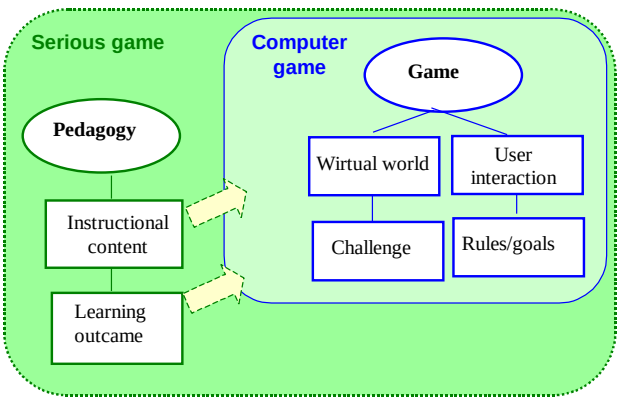

Figure 1. Relation of serious games to other computer games

Since, according to Crawford [5], every (not only computer) game possesses some inherent educational aspect, in the case of Zyda's interpretation, serous game should have clearly specified learning outcome and elaborated instructional content (see Fig 1).

The Serious Games Showcase and Challenge (SGC&C) is an organization, who has the "desire to stimulate industry creativity and generate institutional interest towards the use of digital game technology and approaches for training and education"[6]. Their requirements for what constitutes a serious game when one wants to qualify for their challenge is[7]:

- *"Have clearly defined, measurable learning objectives"*
- *"Provide players with a clearly identified challenge/problem"*
- *"Make use of gaming technology"*
- *"Provide players with positive/negative feedback with respect to progress toward the game's challenge and achievement of learning objectives"*

Garris et. al.[1] find that there are six characteristics that are the core elements for making an instructional game:

1. Fantasy
2. Rules/Goals
3. Sensory stimuli
4. Challenge
5. Mystery
6. Control

The gameflow model, proposed by Sweetser et.al.[8], identifies eight elements which make up the player's enjoyment in games. Some of the elements are closely related to the six elements mentioned in the previous subsection, though some new elements are added as well as some more criteria to follow in the makings of the game.

- *"A challenging activity requiring skill;"*
- *"A merging of action and awareness;"*
- *"Clear goals;"*
- *"Direct, immediate feedback;"*
- *"Concentration on the task at hand;"*
- *"A sense of control;"*
- *"A loss of self-consciousness; and"*
- *"An altered sense of time."*

As it can be seen from this, there is not precise and unique approach how to build serious game. However, above results are very useful if taken into account during design od serious game.

3. REVIEW OF SOME EXISTING GAMES TO TEACH PROGRAMMING

3.1 CeeBot

CeeBot is an example of a commercial game developed by a company named EPSiTEC GAMES [9].

CeeBot is introduced as a game where you have fun while you are learning to program. The language that is used within the game is a language quite similar to C++, C# or Java.

One of the pedagogical concepts included is the development of creativity. CeeBot3 promises that the student will see the game as a tool to be creative and thereby not get the feeling that they are slaves to software used. Furthermore it uses the concept of "*learning through experience*", "*diverting attention from the teaching topic*" and lastly "*The interdisciplinary approach*".

CeeBot offers, among several generalities, an editor and a debugger, which can be seen respectively on figures 2, 3.

Figure 2: The editor in CeeBot[9]

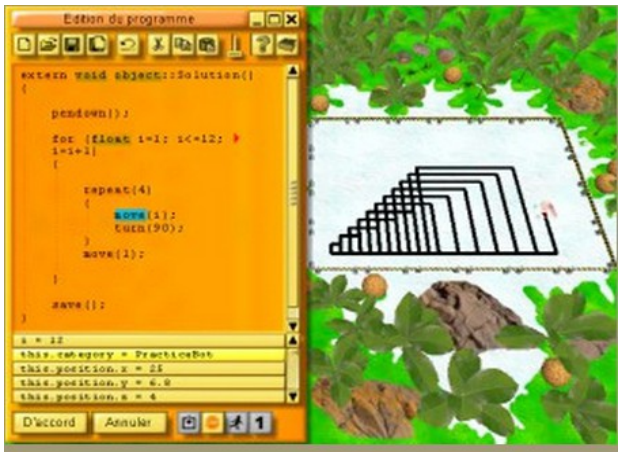

Figure 3: The debugger in CeeBot[9]

3.2 RPG game

Ref [10] describes a serious game for the purpose of teaching programming. The game is within the RPG genre (Fig. 4) and the idea is to programme your main character to solve the tasks that are given. For example, the game begins with the player learning to use variables while entering data about the main character. Whenever the player enters a piece of data he/she has to chose the proper data type.

The concept is to take the curriculum and make it into quests that the player should solve as seen in Figure 5.

Figure 4: Start screen of the RPG game [10]

3.3 Learnmem1

Ref [18] describes the serious game that conveys a subject about computer memory. However, although the game is not about programming, the subjects are closely related, so findings may be applicable for future development.

Figure 6: A screenshot showing the game learnmem1

The game is, as seen in Figure 6, a simple 2D game. The player walks around in rooms gathering knowledge about the subject and he/she then has to answer Y/N or multiple choice questions before being allowed to enter the next room. If the player fails to answer a question he will lose a life and get a hint of what went wrong. Hints are built in the game in order to encourage the player to continue research. Furthermore there are golden books located around the map, that contain all the gathered information.

4. DESIGN AND IMPLEMENTATION OF A GAME TO LEARN PROGRAMMING

4.1 Adopted guidelines for designing a game:

Based on the literature review and analysis of the existing games, we have adopted the following list of guidelines to design a game for initiating in programming.

- *Must have clear rules and goals*
- *Must have clearly defined Instructional content and intended learning outcome*
- *The game shall have leaner gameplay what should convey instructional content*
- *High player sensory stimuli in which sights and sound are different*
- *Challenging game has to match the player's skill (can be done through progressive difficulty level).*
- *Be able to let the player reflect within the game, for instance by suggesting what the newly knowledge acquired should serve them.*

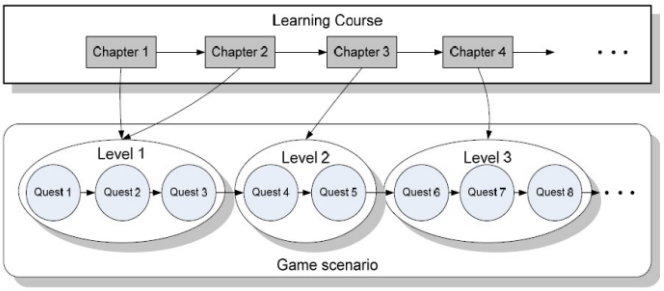

Figure 5 Taking the curriculum and converting it into quests

- *It is recommended that game has some degree of mystery.*
- *Simple mechanics*
- *Must not aim for high end hardware.*

4.2 Instructional content of the game

The instructional content is a knowledge (and/or skill, abilities) which player has to acquire, while the learning outcome is part of that knowledge (skill, abilities) which player has adopted during playing the game.

For the instructional content in the game to be developed, a tutorial from the website cplusplus.com [11] is considered. Its section about variables is what the game should teach. The chosen content from the tutorial contains topics:

- what is a variable,
- identifiers,
- fundamental data types,
- declaration of variables and
- initialization of variables.

These topics are what the players should be able to recall after they have played the game.

4.3 The storyline and graphics design

The story will be about a robot called Tim, who is the protagonist. Tim falls in love with another robot, but in the world they live in, it is illegal for robots to fall in love.

Tims memory is therefore wiped by the memory eaters, but they are not efficient enough and a picture of a beautiful beloved robot remains. It is the goal for the player to restore Tim's memory so he can find his true love again. The idea of the player restoring Tim's memory is inspired by the RPG game concept found in the previous analysis.

As the game is about learning basic programming and the goal is to restore Tim's memory it was decided that the world in where the story unfolds should looks like being inside or close to the RAM blocks of the computer. This means that the background color will be black to give the feeling that the player is inside something closed.'

The game is chosen to be “a 2D side-scroller game", i.e player moves on platforms from left to right (Figure 7).

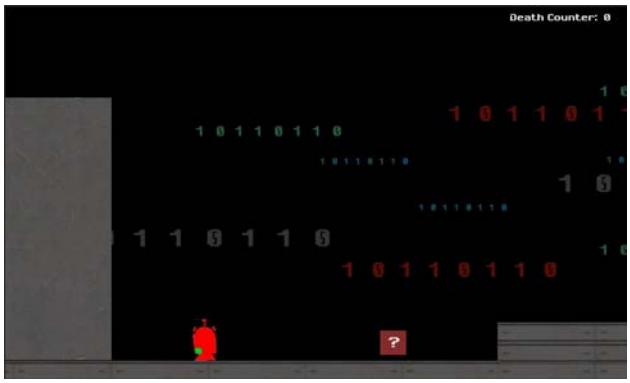

Figure 7 The game environment

Tim (Figure 8) is modelled by a 2D sprite, that looks like a robot. The colours chosen for Tim is red and green, in order to stand out a bit from the rest of the environment.

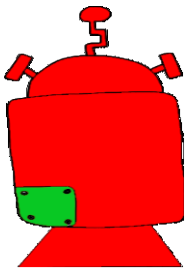

Fig 8: Tim the Robot

4.3 Game dynamics

As earlier mentioned the game will be a 2D side-scroller game and the player is only able to move right or left. For this, the left and right arrow keys are used. The player will also be able to jump using space. The player also have the possibility of opening a menu with escape and enter is used for interactions.

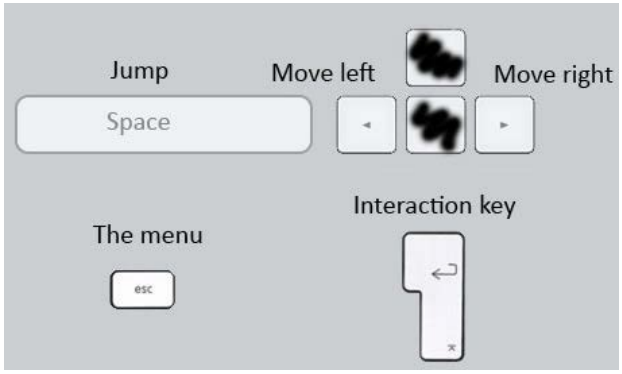

Fig 9: The controls for the game

To progress through the game, player has to solve the puzzles about programming. The player may get the knowledge needed to solve the puzzles from information boxes close to the start spawn point of each level. The information boxes will contain the topics mentioned in start of the design chapter: what is a variable, identifiers, fundamental data types, declaration of variables and initialisation of variables.

The text from the tutorial is rewritten and shortened such that the learning material will fit the story of the game. The box from where the player will get the information will be a red box with a question mark as shown on figure 7 and is used with the interaction key

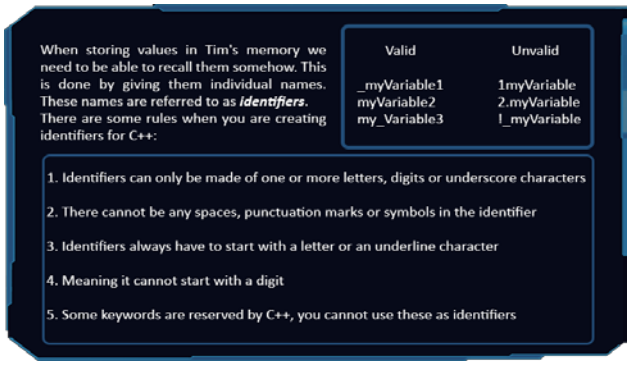

Figure 10: An example of the instructional content of information box

The total of six puzzles has been chosen for the game.

Puzzle 1 – The Tutorial
This level has a task more than the other levels: introducing the controls and helping player to understand the mechanics of the game.

This is also the first time they encounter an information box. Here they are given a few information of from where the C++ starts it execution: the main function. This is how every tutorial starts, and it was found appropriate to place it in the tutorial.

Puzzle 2 – What is a variable
Puzzle two is about introducing the concept of variables. Two new mechanics are introduced in this level as well, the push box and the pole.

Puzzle 3 – Identifiers
In this puzzle the player is introduced to identifiers – the naming of variables. The player learns that there are rules, when making identifiers. After he/she have read the instructional content they will encounter the transparent boxes. The transparent boxes has examples of valid and non-valid identifiers. It is now for the player to choose the right transparent boxes to jump on.

Puzzle 4 – fundamental data types
The players is told the name of the data types and their respective sizes. He/she now have to jump in the right pattern. In this case he/she have to jump from data type to size to data type to size as the player is introduced in the instructional content.

Puzzle 5 - Declaration
The player have to learn how to declare a variable. He/she is able to do this when he/she knows the fundamental data

types and what an identifier is. This appears from the instructional content.

Puzzle 6 - Initialization
In this puzzle there are four push boxes and four holes. Each box belongs to its own hole. The idea is that the player needs to place the correct box in the correct hole. The boxes symbolize a data type, an identifier, an equal sign and a value. If a box goes down in the wrong hole an error will occur and Tim will blow up. This puzzle should make the player to initialize a variable with a value.

4.4 Game implemantation

The game is implemented using Unity3D. Unity3D supports the use of C# and the game is therefore created in the C# language.

The implementation also includes data gathering system that allows collecting test results. After a few different approaches tried, a combination of C# in Unity3D, PHP and MySQL is used for the final implementation.

Detailed description of the implementation may be found in [12].

5. TESTING AND EVALUATING THE GAME

The game accessible through the browser was posted on seven forums where the visitors are interested either in programming or in games, and users were asked for participation in testing.

A total of 24 test persons were registered, among them 22 males and two females. In Figure 11 age distribution of the test participants can be seen.

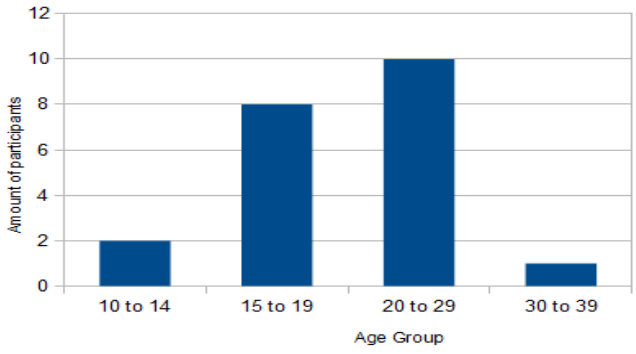

Fig 11: Number of test participants per age

Note. Only the testers who finished the game were taken into account.

Figure 12 illustrates the previous experience the participants had with programming before playing the game.

The participants' motivation was measured on a seven-point Likert scale before the start of the game, between each level and finally after the game.

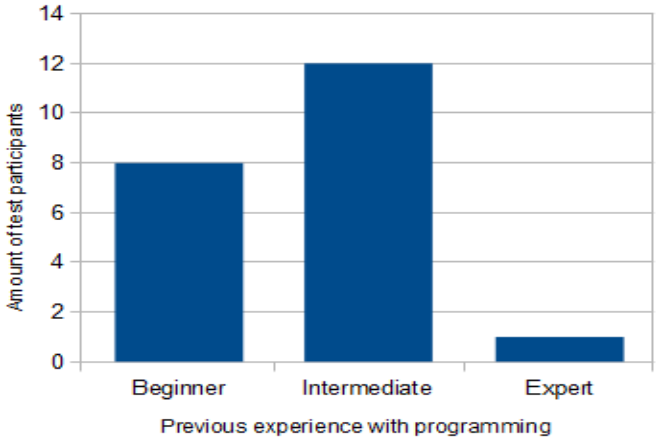

Fig 12: Number of test participants and their previous experience with programming

In figure 13 the average motivation score can be seen. This score was measured asking the participants to what extent they could agree to continue the game, according to the continuation desire theory and questionnaire developed in [14].

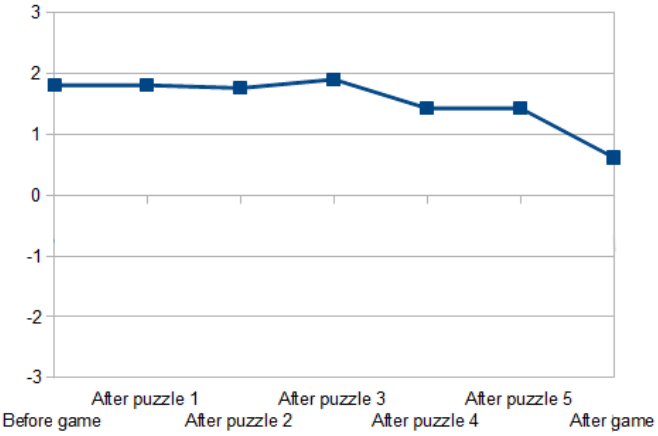

Fig 13: The average motivation score for before the game, the puzzles and after the game

The y-axis has values from '3' to '-3'. Number '3' is where the participant has answered "Agree strongly", '0' is "Neither agree nor disagree" and '-3' is "Disagree strongly".

The average scores for the motivation level in figure 13 indicate that the story/introduction and the three first puzzles are fairly successful.
Puzzles four and five on the other hand do not seem as motivating as the other puzzles and should probably be refined. The possible reason for the near zero scores after the game is that a puzzle game is not a game you would replay, so in fact this result is as expected - once you have completed a puzzle there is nothing more you can do to improve and the puzzle will not be challenging anymore.

To simplify development, implementation did not include sounds/music and beautiful graphics. Some of the testers also commented on the lack of these elements as they certainly would increase player motivation.

To measure knowledge acquisition through the game, the same questions were used before and after playing the game so that a comparison of the results could be done. In figure 14 can be seen the difference of incorrect answer before and after the game.

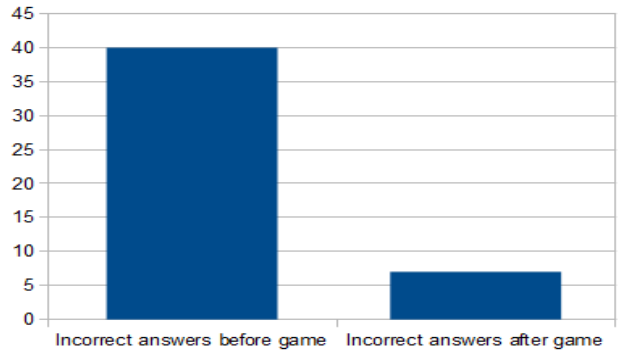

Fig 14: The amount of incorrect answers before and after the game

Before the game there were 40 incorrect answers compared to seven incorrect answers after the game. This is strongly indicating that the game succeeded in learning the player about the chosen elements of C++ programming.

However, it should be noted that only 38% of the participants claimed to be absolute beginners in programming, so these findings may be different for other composition of the testing team.

Generally, looking at the comments from the forums there was a very positive tone in answering the question about overall impression with the game. Some participants really liked the idea. This could indicate that a product of this sort would be very welcomed in the public.

6. CONCLUSION

Guidelines for what is necessary to create a serious educational game that will motivate players to go on are established in this paper. Following these guidelines, a puzzle game that should be able to motivate players in learning basic programming has been created.
The game is implemented and tested in two different aspects. The players were asked to answer about their continuation desire to play the game, in order to check whether the game is motivational enough. At the same time, their actual knowledge about chosen C++ elements was evaluated.
The test results strongly indicate that a serious puzzle game about C++ programming could indeed be motivating for the player to learn about programming.

The motivation score was measured on a seven-point Likert scale ranging from '3' to '-3' - where three is the best - and was recorded before the game, between each puzzle and after the game. The average motivation scores for before-the-game and for puzzles in the first half of the game are rather high (close to '2'), but scores for the puzzles in the second half of the game are close to '1.5' The pre- and post-puzzle knowledge-test results strongly indicate that players are in fact learning something. This is based on 40 incorrect answers before the game compared to seven incorrect answers after the game. However, it should be noted that to draw conclusion whether or not the player had learned anything more test participants with no experience with programming would have to be tested.

LITERATURE

[1] Ahlers, R., Driskell, J. E. and Garris, R.. Games, motivation, and learning: A research and practice model. SIMULATION & GAMING. 33 (4), p441-467.. 2002

[2] Clúa, O. and Feldgen, M., Games As A Motivation For Freshman Students To Learn Programming. 2004

[3] Edgington, J. and Leutenegger, S. (2004). A Games First Approach to Teaching Introductory Programming.

[4] Zyda, M. . From Visual Simulation to Virtual Reality to Games. IEEE Computer Society., p25-32. 2005

[5] Crawford, C. Chris Crawford on Game Design, Prentice Hall, 2003

[6] SERIOUS GAMES SHOWCASE & CHALLENGE. http://sgschallenge.com/wordpress/about/. Last accessed 23th May 2013.

[7] SERIOUS GAMES SHOWCASE & CHALLENGE. http://sgschallenge.com/wordpress/rules/. Last accessed 23th May 2013.

[8] Sweetser, P. and Wyeth, P. GameFlow: A Model for Evaluating Player Enjoyment in Games. ACM Computers in Entertainment. 3 (3). 2005

[9] EPSiTEC GAMES. Available: www.ceebot.com. Last accessed 223th May 2013.

[10] Kataev, A., Shabalina, O., Vorobkalov, P. and Tarasenko, A. (2008). EDUCATIONAL GAMES FOR LEARNING PROGRAMMING LANGUAGES. International Book Series "Information Science and Computing"., p79-83.

[11] cplusplus. (2000). Available: http://www.cplusplus.com/doc/tutorial/variables/. Last accessed 23th May 2013.

[12] Jan Olesen, "Tim the robot - a serious puzzle game about introductory programming", Aalborg University Copenhagen, Graduation project, Medialogy department, Spring 2013

[13] Yadin, A. Reducing the Dropout Rate in an Introductory Programming Course. acm inroads. 2 (4), p71-76. 2004

[14] H. S. Fog, „At the Core of the Player Experience: Continuation Desire in Digital Games.**“,** in: The IEEE Handbook of Games. IEEE, 2013.