This is the published version of a paper presented at *European Conference on Games Based Learning 2017*.

Citation for the original published paper:


Proceedings of the European Conference on Games Based Learning

N.B. When citing this work, cite the original published paper.

Permanent link to this version:
http://urn.kb.se/resolve?urn=urn:nbn:se:miun:diva-31799
Learning to Program by Playing Learning Games

Marie Olsson¹ and Peter Mozelius²
¹Department of Computer and Systems Sciences, Stockholm University, Sweden
²Department of Computer and Systems Sciences, Mid Sweden University, Sweden
marieols@dsv.su.se

Abstract: Game-based learning is an emerging field that has become a part of university education and several researchers describe its strong learning potential. For several subjects there is a rich flora of learning games and commercial-of-the-shelf games available, but for programming education in general and Python programming in particular, the situation is different. There exists an economical aspect as well and to develop learning games for programming education, might not interest the mainstream game industry and most universities do not have the resources needed to develop their own tailor-made learning games. Digital gaming in general has had a fast expansion during the last decade and the generation that now is entering the universities has a habit of regular gaming. At the same time there are several studies reporting about low motivated students in introductory programming courses. In a time when commercial games for programming education are rare, universities must look for affordable ways to construct appropriate learning games. This study has investigated the idea of using learning games developed by university students as additional learning tools in an introductory programming course. Five student constructed games for learning to program in Python have been analysed. Eventually two learning games were selected and tested as part of a five week introductory programming course. The overall strategy has been action research with the aim to improve an existing programming course. A group of students have played the games in two workshops where data has been collected in a mix of questionnaires, observations and group discussions. Findings indicate that the general idea seems to work, but as for all kind of course content, there need to be a thorough assessment and iterative refinement. Despite some found bugs and interface flaws the games had a challenging gameplay as well as learning outcomes. The game-based workshops also had a catalytic effect and created energy as well as curiosity among the participants. To what degree the games and the workshops contributed to concrete learning outcomes will be further analysed after the second examination deadline.

Keywords: game-based learning, GBL, programming education, learning games, Python programming

1. Introduction

Few instructional methods engage similar levels of interest among learners or induce them to persist on tasks for as long as games do. Because of the evident motivational qualities of games, educators and trainers alike seek to use them for instruction (Tobias, Fletcher & Wind, 2014). Computer science students at university level are expected to develop good programming skills during their education and programming courses are often mandatory core subjects.

Programming requires a correct understanding from the beginning and the knowledge is subsequently built up gradually. For that reason, falling behind at the start of a programming course, makes it very difficult to later catch up. Several researchers have also highlighted that learning difficulties in programming courses begins already with the basic concepts (Eckerdal, 2009; van Niekerk, 2016).

Reaching these skills is a challenge for both students and teachers and various forms of teaching has been tested to motivate students and to facilitate the difficult task of learning to program. Software visualisation, on-line learning environments, pair programming and game-based learning (GBL) are some of the practices that have been tested (Braught, Eby & Wahls, 2008; Tobias, Fletcher & Wind, 2014). This study will have a focus on GBL and the use of student constructed learning games.

It is important to understand the effects of games in our learning environments, since it has become a fact that digital games today are played by most of the population (Juul, 2010). The expansion of digital games during the last decade has been fast and today’s generation of university students has developed a habit of regular gaming. The use of games in educational contexts seems to be an interesting way to motivate students (Wiklund & Mozelius, 2013), and game based learning is an emerging field with strong learning potential (Tobias, Fletcher & Wind, 2014).

Previous studies have shown that students become more engaged and motivated in tutoring systems involving game elements (Jackson & McNamara, 2011) and that learning can be increased by the integration of games into curricula (Din, Ward, Chiong, & Shuler, 2001). In the case study Game-based learning and game construction...
as an e-learning strategy in programming education (Olsson & Mozelius, 2016) authors explored in what ways learning could be stimulated in programming education at university level. Furthermore, the article discussed how learning games constructed by the students might be resources for self-learning in introductory programming courses. This study has a focus on evaluating the student constructed learning games in an introductory programming course.

1.1 Problem

Digital games are in the 21st century played by a major part of the population (Juul, 2010) and even to a higher degree by young university students (Mozelius et al., 2016). Research studies have highlighted the strong potential in GBL (Gee & Hayes, 2012; Tobias, Fletcher & Wind, 2014) but it is difficult to find educational games that can be used in programming courses and especially for the Python programming language. Not many commercial-of-the-shelf games fitting in to programming curricula can be found.

In a time when many European universities have to reduce their educational budgets, like at the department where this study is conducted, there is a problem to fund development of tailor-made learning games.

1.2 Research questions

The problem described above could partly be solved by a study providing answers to:

- How could student constructed learning games be reused as resources for self-learning in introductory programming courses?
- Which design factors are important in learning games if they should be able to add value to students' learning processes in introductory programming courses?

2. Related research

Since learning achievements included in a course setting are complex and diverse, many universities apply a student-centred learning approach. Many of today's students, the so-called Net-generation are accustomed to online learning environments and prefer a self-learning approach (Seng & Yatim, 2014). Digital games have the capability to accommodate the essential self-learning environments and can therefore be a suitable tool (among others) to be adapted to the student-centered learning approach (Seng & Yatim, 2014).

The use of digital games in programming education enables students to gain new skills and knowledge through fun and play and to increase their motivation to learn (Prensky, 2005). Some researchers believe that it is because digital games stimulate interaction which can be an essential aspect of the process of learning (Seng & Yatim, 2014). However, to serve as an educational tool, educational digital games need to be designed in a way that promotes learning (Torrente et al., 2010).

Two primary factors that should be met for a pedagogical game to be functional are that the game is: a) useful and functional for educators and b) educational and engaging for students (Marklund, 2014). In addition, Lim (2008) claims that in order to design a game that is meaningful to students and increase their engagement, there are some design principles to consider. One of them is to present a challenge for the player. The above described factors in game based learning with tailor-made educational games (Malliarakis, Satratzemi & Xinogalos, 2013) developed by university students was further investigated in this study.

3. Methods and data collection

Action research has been the overall research strategy. A systematic study was done where action and reflection were combined with the aim to improve an existing programming course. Game based learning with students tailor-made educational games were studied and analysed as complementary learning tools. A case study approach in combination with a content analysis has been used to gather data and the research design is inspired by the book 'Practical research: Planning and Design' by Leedy & Ormrod (2016).

Observations and group discussions were carried out during two workshops announced as extra activities in the studied course. The content analysis meant to systematically and in detail examine the educational value of learning games by identifying patterns, themes or biases, to decide if the games were useful and functional for
Marie Olsson and Peter Mozelius

educators and educational and engaging for students. The content analysis also examined if the games would have a potential to be challenging and motivating for university students.

Data has been gathered in a mix of observations, questionnaires, student postings in online discussion fora and discussions during workshops. All participants in the study were kept anonymous throughout the study following the ethical guidelines expressed by Leedy & Ormrod (2016). The literature review was conducted according to the guidelines in 'Practical research: Planning and Design' with the ideas of investigating related research and discover new aspects of the topics in the study (Leedy & Ormrod, 2016). Main keywords used alone and in combinations to search for articles and books were 'Game-based learning', 'GBL', 'Programming', 'Programming education', 'Learning games', 'Educational games' and 'Python'.

This study is also shared as an open research project in the social networking site for scientists and researchers, ResearchGate:  https://www.researchgate.net/project/Learning-to-Program-by-Playing-Learning-Games

4. Programming courses and the selection of educational games

The fundamental idea for this study is to select some of the learning games that were constructed by the students in the course described in 4.1, for reuse as resources for learning basic programming concepts in the course described under 4.2. Multimedia programming in Python is a distance course given for students all over Sweden with earlier experience of programming, but not necessarily in Python. The course is a summer course and a voluntarily choice for students that want to extend their programming proficiency. Students are normally motivated and also with excellent pre-knowledge.

An introduction to programming in Python is the first mandatory course for students in the first year on a Bachelor’s programme. In the studied course batch half the student group comes from a programme for Interaction design and the other half from a programme on Digital media. What they have in common is the curiosity for building things, but also the lack of pre-knowledge in programming and the more general computational thinking. Students’ motivation has large variations and it is frequent that students get stuck and frustrated in the initial course phase.

4.1 Multimedia programming in Python

Almost the entire syllabus is built around the idea that multimedia programming in Python could joyfully be learnt with GBL and by game construction. Lectures, online tutorials and the course literature all have a common focus on analysis and synthesis of digital games (Olsson & Mozelius, 2016). Practically all assignments are based on game construction or creating game mechanics, or parts of games with graphics and multimodality. Initially, game ideas are basic to get students started, but later assignments have an increased complexity. Graphical user interfaces and user interaction are initially created with the Tkinter library that is a part of the Python standard installation. For the final assignment and the learning game project most students build their games with the use of PyGame, where sound effects and music can be handled smoothly and platform independent.

Multimedia programming in Python is given with a 50% study pace as a summer course from early June to late August. Out of initially about 400 applicants the intake was 90 students, but not more than 40 students actually started the course in 2016. Eventually 30 students completed all their assignments, and only 21 students completing all assignments and the learning game project to the second deadline in November 2016. With a larger student group like some years ago the set of learning games would have been larger but not necessarily of higher quality. The submitted learning games have a technical standard that is the highest ever in any course version. Average grades are also the highest ever with one third of the students meeting the criteria for the highest grade of A for their learning game projects.

4.2 An introduction to programming in Python

In the first part of the course students are introduced to the traditional procedural programming techniques that are necessary in all modern programming languages. Important concepts to learn are variables, data types, selection, iteration, Boolean conditions, functions and function calls, and data structures to be able to build solutions to the course assignments. In the second course part students build graphical user interfaces with the use of the Tkinter module. The second half of the course also introduces the students to programming techniques such as error handling, file handling and basic code testing. Examination consists of five programming
assignments covering the practical programming skills and a written exam assessing students' theoretical programming knowledge.

An introduction to programming in Python is given in blended mode with recorded lectures (4), recorded lessons (3) and partly recorded workshops (3). The idea with the workshops is to facilitate for students that get stuck in the inception phase with practical training on selected concepts that students tend to find hard. In the studied course instance the workshop themes were: Data types, Syntax errors and Functions. Three concepts authors earlier have experienced a tendency to student problems. Data types was the topic for Workshop 1 with the use of the game PyMem and syntax errors was the theme for Workshop 2 where students played the game PyBomber.

Both games are described more in detail below in 4.4 and 4.5. Authors were not able to find any game on functions and function calls with arguments and return values, but not all workshops have to involve learning games. An important part of the workshop design was the choice to start with a teacher-led introduction to the learning game themes with real-world Python examples.

### 4.3 Game selection process

Learning games were initially analysed from a set of project assignments from two courses, where one is the course described in 4.1. The other course was a course on Game-based learning with a similar project assignment where students could chose to build learning games on Programming, IT-Security or on History. Several games from the GBL course had interesting backstories and catchy gameplay, but none of the programming games was based on the Python programming language.

Firstly, all learning games on programming from the multimedia course were analysed and compared. All games except one had a comprehensible gameplay seen from a novice student perspective, and there were some games with identical themes, like two Memory hybrids and two games on data types. The first choice was PyMem since it had more advanced technical design than the other Memory game, with features for altering the number of tiles and files where teachers can add new data type categories and data type examples. The other game on data types had sophisticated graphics with an Asteroid-like design, but none of the authors' managed to handle the complicated interface. The second choice was PyBomber with the creative idea of integrating the Python standard editor in a bomb throwing clone of the classic digital game Snake.

### 4.4 Game 1 - PyMem

![Figure 1: A screenshot from the game PyMem](image)

The first learning game that was tested in the study is called PyMem, and was developed by a male student during a summer course on multimedia programming in 2016. The course and the student projects are described
in detail in 4.1. With the idea from the authors that data types are even more important to highlight in a dynamically but strongly typed language like Python, the first workshop and game taught data types. The data types to match in the game were: integer, float, string, Boolean, list and complex.

PyMem is a variation of the well-known Memory game, but instead of matching two identical items, various basic data types in the Python programming language should here be matched with exemplifications of the data type. For example, a tile with String should be matched with a tile with "Hello World!" Thus, the main purpose of the game is to improve data type skills of the players.

The exemplifications of the data types are changed randomly during the game, which increases the level of practicing data types during the game. The data types and their exemplifications can be changed by updating the text files in the program code and the size of the Memory grid can be adjusted by the player. Feedback is also provided when the data types do not match.

4.5 Game 2 - PyBomber

The second learning game that was tested in the second workshop is called PyBomber. This game was also constructed as a project assignment during a summer course on multimedia programming. In PyBomber, the player acts as the Python syntax snake that in an IDLE environment should find syntax errors and bomb them away. IDLE is the basic Python editor that is included in the Python standard installation. The player must keep a safe distance from the bomb blasts to stay alive in PyBomber.

After a successful bombing of all errors in one level the player will find more challenging syntax errors at the next level. PyBomber also have a built-in timer measuring how fast players can bomb away all syntax errors. The didactic idea behind the game is that syntax errors are a hampering obstacle for students new to programming and that they in some way need to be bombed away. The game was built as a project assignment of a female student in a summer course on Multimedia programming in Python.

Figure 2: Screenshot from the game PyBomber
5. Findings and discussions

5.1 Observations and group discussions

PyMem

In the first workshop where the game PyMem was tested, students were very engaged, and most of the students continued to play the game over and over again. Some students asked questions about the data types that they were uncertain about, to be able to complete the game. Many students also cooperated while they were playing and gave comments and suggestions how to improve the game. Some students were happy to complete the game in as little time as possible, while others thought it was more important to spot the data types. Most surprising observation was that some students opened up the source code and tried to understand how the game mechanics are built.

PyBomber

In the second workshop where the game PyBomber was tested, students were quite engaged in the game and cooperated to complete the game. Time was a crucial factor to complete the game since there were time constraints on each level. We observed a greater need for collaboration, in order to find the errors in the code before time expired. In the group discussions some students asked questions about Python syntax, to be able to complete the game and they also commented on the game. Two students were observed trying hard to complete all levels of the game, but got stuck in the last level. When they couldn’t find the error, they didn’t want to give up the game. The same students also wanted to complete the game in as little time as possible. Most surprising observation was that a student found PyBomber engaging to the level that it brought back the compulsive gaming that she was trying to get rid of.

5.2 Questionnaires

PyMem

A questionnaire consisting of questions about students’ earlier experiences of learning games and games in general, opinions and perceived usefulness of the learning game PyMem was handed out. There were also questions about what students perceive as important aspects of learning games. The questionnaire were handed out to a total of 43 students, where 33 of them replied that they had no earlier experience of programming in Python. 9 students stated that their earlier experience of Python were poor. 33 of the respondents answered that they were never playing learning games and 10 respondents said that they played learning games about once a month.

The analysis of the questionnaires reveals that the major part of the students thought that the game PyMem provided valuable assistance in learning Python data types. Among the students who answered the questionnaire, 27 rated the game as quite helpful when practicing the data types, and 6 students rated it as very helpful. On the question to what degree the game PyMem gave a better understanding of Python data types, 21 respondents answered that the game helped quite much to understand data types better.

To learn from the game was the most important aspect of learning games in general, according to the questionnaire responses, where 28 of the respondents marked the alternative most important, on the five item scale ranging from extremely unimportant to most important. Only 5 respondents thought that learning from the game was of average importance or less important. The second most important aspect was the quality of learning content. 21 respondents marked the quality of learning content as most important and 18 respondents marked it as important.

Easy to understand the game was the third most important aspect of learning games in general among the students, where 34 respondents altogether marked this aspect as most important or important. Most students marked the aspects visual appeal, easy to play and fun to play as important or average important, which was a somewhat surprising finding. The least important aspect regarding learning games, in the answers was the importance of winning the game. 27 of the respondents stated this aspect as average important or unimportant.
PyBomber

For the second workshop, a questionnaire was constructed in the same style as for the first workshop, but with a focus on opinions and perceived usefulness of the learning game PyBomber. There were also questions about students’ own ideas to further develop the use of learning games in programming education. The questionnaire were handed out to a total of 22 students.

Practicing Python syntax with the game PyBomber was appreciated by most of the students. 12 students were of the opinion that the game was a quite helpful tool when practising Python syntax and 4 students felt that it was very helpful. 12 of the respondents thought that the game gave them a better understanding of Python syntax, which is over half of the students and a surprisingly positive result.

Time constraints and not having to start all over from the beginning when failing a level in the game, were dominant inputs from the students regarding ideas to further develop the use of learning games in programming education. Two of the respondents expressed the time pressure in the following ways: “Time pressure causes insufficient focus” and “In this particular game, it would be better if you had more time, or that there was no time limit at all (if you became too stressed, you could just click around the screen and accidentally get right without knowing why)”

To be able to get help or hints was another suggestion for improvement, as well as to write self-composed code in the game. One respondent wrote: “Make a game where we get to write code ourselves. I would learn a lot by practicing to write myself”

5.3 Game analysis

Both games were chosen with the idea that the well-known game ideas would not need any introduction in the workshops. Practically all students have earlier played some variations of Snake and Memory game, and there was no need for time consuming introductions of the game ideas in the workshops. This was important since workshops also must include some theoretical input and time to play and replay the games.

The main reason for choosing PyBomber was the unique and creative idea of having the bomb armed snake integrated in the IDLE environment, crawling around in authentic Python code. There was no similar project submission and syntax errors are always problematic in programming languages. Especially in Python, where small things like a missing colon or an incorrect indentation can make otherwise correct code impossible to execute.

PyMem had several competitors and some of them definitely with more sophisticated graphics and more challenging gameplay. There were some usability flaws, but the flexible features where not only the tile grid, but also, and more important, the learning content could be altered without actually touching the code. This is a feature that makes the game useful even for teachers with less Python knowledge, enabling that Python data types must not necessarily be the game topic.

Finally, we find it congenial that games designed to learn Python also are designed in Python and that providing the source code can be a way to make students interested in more advanced programming techniques. PyMem and PyBomber can be seen as trivial mini-games from the players’ perspective, but it takes effort as well as quite complex programming techniques to build them.

6. Conclusions

As pointed out in a study by Gomes & Mendes (2007), programming education has a tendency to teach dynamic concepts through dry and static materials. Authors' believe that learning to program by playing learning games could be a way to introduce fundamental and important programming concepts, but they better have to be combined with other learning activities. The idea of using games with relatively short gameplay and clear learning outcomes might be a way to achieve what Johan van Niekerk (2016) has promoted as Brain-Compatible Learning, where explaining relatively small chunks of important knowledge, can later facilitate students more complex learning.
The answer to the second research question about important design factors for learning games in introductory programming courses, authors find the most important to be:

- **Focused learning outcomes**, it is good enough to learn a small well-defined topic in each learning game.
- **Engaging gameplay**, a boring non-challenging gameplay could rather decrease learners’ motivation (Brusse, Neijens & Smit, 2010).
- **Relatively short gameplay**, the gaming should be an integrated part of a 2 hour learning session for this kind of curriculum integrated extra activities.
- **User-friendly design**, the game should be easy to understand and with a self-explanatory game idea.

On the other hand, authors’ have also find COTS games on learning to program with a longer gameplay and wider focus, as alternatives to consider. One example of a learning game worth further investigation is TIS 100, developed by Zachtronics Industries (TIS-100, 2016). A puzzle game where the player learns assembler programming. A similar game teaching Python would be interesting for this course, but rather as an extracurricular activity.

### 7. Future work

The next immediate step to further validate the findings in this study is to analyse the evaluation questionnaires and the written exams for the course Introduction to programming in Python. This can be done after the second deadline for the course in March 2017. If the tested games get a positive feedback and if the chosen workshop topics seem to have had an impact on the answers to the written exam, it would be reasonable to make a new iteration and try the concept in another student group.

In a longer perspective it would be interesting to repeat the game construction project with more specific instructions, focusing on identified thresholds in introductory programming. To get a better view of which programming concepts students find to be the initial thresholds, there will also be questions added on this subject in the course evaluation questionnaire.

### References


