Programming Education: Using the Raspberry Pi and games to teach programming

Martin D Flanagan
University of Cape Town
Rondebosch, Cape Town, South Africa
FLNMAR011@myuct.ac.za

ABSTRACT

Programming can be really difficult to learn, given that you need to learn a programming language and programming constructs at the same time. Most, if not all first year Computer Science students are forced to take a course in programming. Students can often find it difficult to learn to program as some of them would not have taken a programming class before coming to a tertiary institution. Students tend to lose interest very quickly and can become demotivated when they find things too difficult and may even drop the course or major. Games are often thought to be a form of entertainment, but it can be incorporated into projects to make it more fun and engaging. It can also be used as a form of motivation (as an incentive). Microcontrollers are quickly becoming a part of our lives, with all these smartly connected devices which we can monitor with a single app on our phone, to having a smartly connected home system. Microcontrollers, if used correctly, can also be used to enhance a student's learning experience. This literature review aims to find ways to make learning programming more creative and intuitive for these first year students.

CCS CONCEPTS

• **Applied computing → Interactive learning environments**; • **Hardware → Sensor devices and platforms**; **Sensor applications and deployments**.

KEYWORDS

Computer Science, Education, Games, Raspberry Pi, Programming, Python, Panda3D, 2D and 3D programming

ACM Reference Format:
Martin D Flanagan. 2020. Programming Education: Using the Raspberry Pi and games to teach programming. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Computer Science is a huge field and is still growing, as such, many students enroll in these courses to take advantage of the fact that programming is a good skill to have and will most likely be needed in most jobs today. Most of the students who take a programming course are Computer Science students, but the other students are normally from other faculties which prescribe a programming course for some of their students.

Ozoran et al. [14] says that programming is one of the subjects that challenges students the most [2, 7, 14], this seems to be the case with most of the literature and as Gomes et al. [9] mentioned: "high failure levels are common in initial programming courses". It could be the case of the courses being too difficult, or the students are not grasping the content as the lecturer(s) would expect them to.

This literature review aims to look at the current ways that students are learning programming, how they are taught and how they learn through assignments and the relevant texts and also how using games in the assignments engages the students and helps them to learn the material. The various programming languages which is taught in first year CS will also be looked at.

2 EDUCATION IN CS

The main focus of an introductory course to programming in computer science is mainly to develop a student's problem solving ability [9], but this does not seem to be the case in the literature. Fee et al. [6] notes that only a few students who take such courses want to improve their problem solving skills and also that students only develop an appreciation for problem solving later.

The requirements for these CS courses are normally critical thinking, a general idea to problem solving and computational thinking among others [6, 17]. A lot of the students do have these skills, but still struggle learning programming.

For the most part, the students seem to struggle learning the basic concepts because of its complexity [2, 14], ranging from the basic ideas of variables to the complex idea of an array (or lists) and loops. According to Ozoran et al. [14] these can become barriers for learning programming. They also go on to say that this can also be due to learning a high number of concepts in such a short time and hints that maybe the courses need to span over a wider range of time.

The students tend to learn more about the syntax than about solving the problem [14, 17] and some go as far to say that they worry more about getting the program to run and resorting to trial and error [2, 14]. This defeats the purpose of learning the concepts of the course.

One solution to this problem is using Scratch as the programming language [14] and another solution could also be using the Greenfoot environment [11] but both environments are a drag and drop based environment (the code structures are already there in pre-made blocks that just need to be dragged and dropped).

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

A good solution would be to use Problem Based Learning (PBL) [6]. By doing this, we can set up the problems in such a way that the students can relate their own experiences to the problem.

3 GAMES IN COMPUTER SCIENCE EDUCATION

Using games as a form of programming education is not new, in fact many students who enter university already have experience with games [5, 12, 18], whether it be playing or creating them. Games can be used to teach programming and can motivate the students while they are creating and playing them [5, 7, 12, 18].

Vahldick et al. [18] states that there are two approaches used when you include games in Computer Science education: creating and playing games. The creation aspect boils down to coding a small game to apply the programming concepts and in the second approach, "the students play games to reinforce and practice concepts and programming skills" [18]. They also discovered that the genre of suitable game types would be puzzle games, games where simulation is involved and strategy games among others.

At the University of Denver [12], they offered a Game Development undergraduate degree where they taught the basics concepts of Computer Science (CS) in the first quarter by focusing on a game (in their case which was a ball moving around on a screen). The other 2 successive courses were also taught with a game focus. The point of these courses was to keep the students engaged as well as to help the students learn the concepts of CS. Looking at the results of their surveys, the students enjoyed the game focus approach and their retention of information through the year improved because of this.

Another study [7] revealed that introducing a game-based approach increased the success rate of the students (and the pass rate) and even decreased the amount of students dropping the course. They also focussed on a different learning style where students would interchange between acting as a designer and a programmer. Their syllabus started teaching small constructs and the assignments grew to become a more complete game application.

4 2D VS 3D PROGRAMMING AND THE 3D ENGINES

It could be tough to decide whether a 2D (2-Dimensional) game or a 3D (3-Dimensional) game is better in terms of how it can be used to educate students. Ak et al. [1] states that an educational computer game can be created in a 2D or a 3D format only. Ak et al. [1] conducted a study where they compared different learning environments (2D, 3D and traditional) to each other. They used the same game mechanics across all the groups to keep the study as fair as possible and they found that the learning gains for all 3 learning environments were high and had no significant difference and in the end, they noted that students are more likely to go for the 2D games because the students value it more [1].

Coding a 3D game or even a 3D project can be difficult and one might need a graphics engine to handle all the graphics processing. A game engine provides abstraction by "hiding complex concepts and providing powerful resources to the developers" [15].

Panda3D [10, 13] is an open-source, free-to-use engine that can be used for 3D games. It uses both Python and C++ programming languages [13] but it is simply Python code that is used on the developing end. It is compatible with a large range of graphics cards [10] and runs on Windows, Mac and Linux [10, 13, 15]. It is also able to work with low-end display devices [10].

Alice is a 3D interactive graphics programming environment which is built on the Python programming language with the aim of making it easier for beginners in programming to develop 3-dimensional environments [4]. The main idea behind Alice is that programmers will be able to control and manipulate objects in a virtual 3D environment with input from the mouse and keyboard connected to the computer [4]. Cooper et al. [4] also answers an important question about how much a programmer needs to know about 3D graphics and animation, they found that only an understanding of the coordinate system and the spatial relationships between the objects in the environment.

Dickson et al. [5] notes that when setting up a game development course, among others, some things to keep in mind are:

- when choosing the game engine, it needs to have a good developer community and documentation,
- the students need to be familiar with the programming languages used, and
- the game engine needs to support cross-platform development.

Panda3D and Alice seem to tick all those boxes.

5 SINGLE BOARD COMPUTERS

Single board computers and microcontrollers have become a part of our daily lives even if we do not know it. A lot of these boards are being used to smartly 'connect' our devices in our homes.

The Arduino board is an example of a microcontroller. It has an open source platform [3, 16] and it has a big user community as well [16]. Rubio et al. [16] conducted a study where the board was used to teach various concepts, from variables to loops and also showed how engineering 'breadboards' can be used together with this board to teach these concepts. A loudspeaker was used to teach the concept of arrays (lists), while conditions (if, if else statements) were taught by capturing data from the light sensor and the concept of looping was taught by continuously grabbing information from a sensor and this was all done with a breadboard and other electrical components connected to the Arduino board [16]. Rubio et al. [16] also claims that the Arduino was designed with the aim that artists and designers would use them.

The Raspberry Pi (RPi) is a "low-cost, credit card sized computer" [8] that is just as powerful as a normal/standard computer and can be used for programming purposes [8, 11, 19, 20]. It is also described as "easily affordable for children" and a "program-oriented device" [11]. It has been used to teach the course "Introduction to the Internet of Things " at Indiana University Purdue University Indianapolis (IUPUI) [20]. There they used the Raspberry Pi and some sensors to create multiple projects including a "Raspberry Pi Stock Ticker". Wirth et al. [19] notes that the RPi runs an operating system, thus acting like a mini-computer, while the Arduino is just a microcontroller.

6 DISCUSSION

Programming can at some points be very taxing to learn [2, 14], provided if the person has not come across the concepts before or has not sharpened their critical thinking skills [6]. It's not just the concepts that need to be learnt, but also a whole new coding language, in most cases, its syntax and all the other 'nuts and bolts' needed to be proficient in the language. The students tend to get caught up in learning the syntax of the language [14, 17] that their problem solving ability might not be tested and they will not improve this by doing so. This could also mean that possibly they need to build on their problem solving ability by practicing the concepts in a simpler programming language where they do not have to worry about whether the program will run or not [2, 14]. So instead of practicing these constructs in Java, they could possibly do it in a programming language like Scratch or Python which is simpler. By doing this it would mean that they build up their problem solving ability and will have a better understanding of the concepts as they will have more time to focus on it.

6.1 Games Discussion

One solution to the problem seems to be including games in the course projects [7, 12] to help the students to retain the information that they learn [12]. By including games in the projects of the course it can not just motivate the students, but also allows them to see what the code that they write can do and to check whether they have made a mistake while coding. This may be something small, but can have a big impact on how the students learn a programming language.

Offering different projects would give the students room to choose the project that they want and to allow more freedom and room for creativity as one programming problem can have many solutions that all end up with the exact same answer. This also allows the students to explore multiple ways of problem solving as they have the room to plan and decide how they will tackle the problem.

6.2 Hardware Discussion

Another solution would be using a microcontroller or a single board computer to educate the students. Both the Arduino board and Raspberry Pi were very successful in helping students learn some programming concepts [3, 8]. When comparing the two boards, it should be noted that the Arduino is just a simple microcontroller, while the Raspberry Pi runs its own Operating System (OS) [19]. While both boards offer connection to external devices and sensors, the Raspberry Pi is the better board to have. Although both boards are small enough to carry around and lightweight, the Raspberry Pi has the resources needed for coding and the space needed to install the necessary frameworks. The reason it stands out from the Arduino is because it is a pocket-sized computer, not a simple microcontroller.

6.3 2D and 3D Projects Discussion

Students prefer doing the game based assignments as opposed to normal programming assignments [7, 12] and even though most students would prefer doing 2D assignments [1], it makes sense that at least one assignment includes some form of 3D coding so that the students can challenge themselves. By having 2D and 3D assignments, the students will be able to see the effect that their code has inside a virtual environment [4]. By adding an extra dimension, we allow the user (students in this case) to see the virtual environment from a different perspective.

Alice is ideal for new aspiring programmers in the sense that it is easy to use and does not require much understanding of the animation and rendering of the 3D environment. When compared to Panda3D one will notice that Alice is meant to be used for teaching the basic concepts of programming and 3D spaces, as Panda3D can do much more than just simple shapes and 3D environments. Keeping in mind that Panda3D is a game engine and Alice is a graphics programming environment, they do fit the description which Dickson et al. [5] was looking for in potential engines to use in a game development course.

Both Panda3d and Alice are free to use but requires the developer to know some Python programming, which makes it ideal for new students learning to code. Both are also available cross-platform which means that if it is selected, the students will not need a specific operating system to use the engine and its framework, which can make both good options to use as an engine.

6.4 Programming Language Discussion

While looking at the literature we have come across Scratch [11, 14], Python[4, 10, 11], Java[4, 11] and C++ [12, 13, 15]. They are all good programming languages, but there are big differences with each of them. If we were to break down the famous 'hello world' programs for some of these, we see the following:

```
In Java:
    public class ClassName{
        public static void main(String [] args){
            System.out.println("Hello world.");
        }
    }

In C++:
    #include<iostream>
    int main(){
        std::cout << "Hello world." << std::endl;
        return 0;
    }

In Python:
    print("Hello world.")
```

In Java the students will have to understand the constructs of a class and a main method, while in C++ the concepts of a main method and importing needs to be understood. With Python the students can get coding right off the bat, which makes it less dificult for the students to get started on Python coding.

It is evident that programming language of choice would be Python as it is simple to learn and code and does not require that students remember constructs of classes and objects. Java and C++ would not be ideal in this case as the students would have to worry about classes, objects, inheritance and so on. The reason

why Scratch and the Greenfoot environment are not suitable to use is because all the code is in a drag and drop structure and as we want to focus on the students learning the CS programming concepts and a programming language side by side, using this kind of programming environment is not the best way to do it.

7 CONCLUSIONS

Programming can be difficult to learn and understood and while the usual approach of learning by solutions can be effective, it can also be boring and lead to students becoming discouraged and uninterested in the material. Another problem is that due to the content of the course being taught in such a short space of time, students are already under a lot of pressure and because they have to learn a programming language alongside the other course content, they often become more focused on getting their program to run than actually learning the concepts.

One solution is introducing PBL (Problem-Based Learning) where the problems can be made so that students can relate to the problem in some way, this will engage the students and hopefully encourage and motivate the students to do the assignments and in turn will learn the content better and retain the information that they learn along the way.

Another solution is to integrate games into the courses such that it can motivate the students as well as to keep the interest of the students on the subject by giving them game-based assignments where they can get visual feedback of what they are doing. They can use the visual feedback to determine whether they are on the right track to solving the problem as they will be able to determine whether the correct thing is being done in the correct moments. The games will also make the projects more fun and could also be used as an incentive to complete the project (i.e. the student would want to complete the game so that they will be able to play it and have the achievement of completing it).

A third solution would be to incorporate microcontrollers or single-board computers into the curriculum such that the programming assignments include using some of the features of the boards and by using some external parts like a breadboard with sensors or potentiometers as an input source instead of the normal keyboard input. External boards that plug into the boards could also be used as a source of input, such as the Sense-Hat for the Raspberry Pi.

Both 2D and 3D game development are good options for education based games, but given that students value 2D game projects much more than 3D game projects, it might be a bit difficult to justify having 3D game projects only. Having a list of 2D and 3D projects that students can choose from will solve the issue, but it should be fair on the students too (i.e. The student that takes a 2D project has the same amount of work that needs to be done as the student taking the 3D project).

Using a graphics engine makes the development so much easier and can remove all taxing development from the rendering and model generation, so that a developer can focus on the aspects of the game instead of the sheer volume of code needed to render these 3D and even 2D environments. Panda3D stands out as the only one that fits this project's needs at this time as Alice is not a game engine so it does not meet the requirements for this project.

This project aims to use the Raspberry Pi with any external add-ons needed and include some game-based assignments to teach simple programming concepts.We also aim to motivate and engage students in the coursework. Both 2D and 3D projects will be used as it can give a different sense of perspective and Panda3D will be the graphics engine used with Python being the underlying programming language.

REFERENCES

[1] Oguz Ak and Birgul Kutlu. 2017. Comparing 2D and 3D game-based learning environments in terms of learning gains and student perceptions. *British Journal of Educational Technology* 48, 1 (2017), 129–144. https://doi.org/10.1111/bjet.12346

[2] VH Allan and MV Kolesar. 1997. Teaching computer science: a problem solving approach that works. *ACM SIGCUE Outlook* 25, 1-2 (1997), 2–10. https://doi.org/10.1145/274375.274376

[3] Arduino. 2020. *Arduino - Introduction.* Retrieved Apr 24, 2020 from https://www.arduino.cc/en/guide/introduction

[4] Stephen Cooper, Wanda Dann, Randy Pausch, and Randy Pausch. 2000. Alice: a 3-D tool for introductory programming concepts. In *Journal of computing sciences in colleges*, Vol. 15. Consortium for Computing Sciences in Colleges, 107–116. https://dl.acm.org/doi/pdf/10.5555/364133.364161?download=true

[5] Paul E Dickson, Jeremy E Block, Gina N Echevarria, and Kristina C Keenan. 2017. An experience-based comparison of unity and unreal for a stand-alone 3D game development course. In *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education*. 70–75. https://doi.org/10.1145/3059009.3059013

[6] Samuel B Fee and Amanda M Holland-Minkley. 2010. Teaching computer science through problems, not solutions. *Computer Science Education* 20, 2 (2010), 129–144. https://doi.org/10.1080/08993408.2010.486271

[7] Maria Feldgen and Osvaldo Clúa. 2004. Games as a motivation for freshman students learn programming. In *34th Annual Frontiers in Education, 2004. FIE 2004.* IEEE, S1H–11. https://doi.org/10.1109/FIE.2004.1408712

[8] Raspberry Pi Foundation. 2020. *What is a Raspberry Pi?* Retrieved Apr 24, 2020 from https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/

[9] Anabela Gomes and António José Mendes. 2007. An environment to improve programming education. In *Proceedings of the 2007 international conference on Computer systems and technologies*. 1–6. https://doi.org/10.1145/1330598.1330691

[10] Mike Goslin and Mark R Mine. 2004. The Panda3D graphics engine. *Computer* 37, 10 (2004), 112–114. https://doi.org/10.1109/MC.2004.180

[11] Michael Kölling. 2016. Educational programming on the Raspberry Pi. *Electronics* 5, 3 (2016), 33. https://doi.org/10.3390/electronics5030033

[12] Scott Leutenegger and Jeffrey Edgington. 2007. A games first approach to teaching introductory programming. In *Proceedings of the 38th SIGCSE technical symposium on Computer science education*. 115–118. https://doi.org/10.1145/1227310.1227352

[13] Carnegie Mellon University. 2020. *Panda3D–Free 3D game engine.* Retrieved May 10, 2020 from http://www.panda3d.org/

[14] Dincer Ozoran, N Cagiltay, and Damla Topalli. 2012. Using scratch in introduction to programming course for engineering students. In *2nd International Engineering Education Conference (IEEC2012)*, Vol. 2. 125–132.

[15] Rafaela V Rocha, Rodrigo V Rocha, and Regina B Araújo. 2010. Selecting the best open source 3D games engines. In *Proceedings of the brazilian Symposium on Games and Digital Entertainment, Florianópolis, Santa Catarina, Brazil.* https://pdfs.semanticscholar.org/40ca/2f43ea66fde72040cce18dfed2718b7a45ba.pdf

[16] Miguel A Rubio, Carolina Mañoso Hierro, and APDM Pablo. 2013. Using arduino to enhance computer programming courses in science and engineering. In *Proceedings of EDULEARN13 conference*. IATED Barcelona, Spain, 1–3. https://pdfs.semanticscholar.org/c722/2f0f4b60735ac62bafd9fe17312657983526.pdf

[17] Damla Topalli and Nergiz Ercil Cagiltay. 2018. Improving programming skills in engineering education through problem-based game projects with Scratch. *Computers & Education* 120 (2018), 64–74. https://doi.org/10.1016/j.compedu.2018.01.011

[18] Adilson Vahldick, António José Mendes, and Maria José Marcelino. 2014. A review of games designed to improve introductory computer programming competencies. In *2014 IEEE frontiers in education conference (FIE) proceedings*. IEEE, 1–7. https://doi.org/10.1109/FIE.2014.7044114

[19] Michael Wirth and Judi McCuaig. 2014. Making programs with the Raspberry PI. In *Proceedings of the Western Canadian Conference on Computing Education*. 1–5. https://doi.org/10.1145/2597959.2597970

[20] Xiaoyang Zhong and Yao Liang. 2016. Raspberry Pi: an effective vehicle in teaching the internet of things in computer science and engineering. *Electronics* 5, 3 (2016), 56. https://doi.org/10.3390/electronics5030056