**College of Computing Technology**

**An investigation into the use of a new learning tool for BSc in IT 1st year students**

**Richard Loy SB12407**

**May 2016**

**Supervised By: Mr Graham Glanville**

Table of Contents

[**1.** **Abstract** 2](#_Toc449904104)

[**2.** **Focus of Study** 3](#_Toc449904105)

[**3.** **Literature Review** 4](#_Toc449904106)

[**4.** **Research Design and Approach** 9](#_Toc449904107)

[**4.1** **Research Questions** 9](#_Toc449904108)

[**4.2** **Research Methodology** 9](#_Toc449904109)

[**4.3** **Research Methods** 10](#_Toc449904110)

[**4.4** **Ethical Review** 11](#_Toc449904111)

[**5.** **Data Collection & Analysis** 12](#_Toc449904112)

[**6.** **Lessons Learned** 13](#_Toc449904113)

[**7.** **Conclusions and Recommendations** 15](#_Toc449904114)

[**8.** **Reference List** 16](#_Toc449904115)

[**9.** **Appendices** 18](#_Toc449904116)

[**9.1** **Appendix 1: Manual** 18](#_Toc449904117)

[**9.2** **Appendix 2: Systems Analysis** 22](#_Toc449904118)

[**9.3** **Appendix 3: Systems Design** 31](#_Toc449904119)

[**9.4** **Appendix 4: Implementation of the system** 36](#_Toc449904120)

[**Hardware and Software Assembly** 36](#_Toc449904121)

[**Manual for Creating the Raspberry Console** 36](#_Toc449904122)

[**Videos for Creating the Raspberry Console** 37](#_Toc449904123)

[**Games** 37](#_Toc449904124)

[**9.5** **Appendix 5: Student Presentation** 53](#_Toc449904125)

[**9.6** **Appendix 5: Mind maps** 55](#_Toc449904126)

1. **Abstract**

This project is an attempt to introduce a new learning engagement tool for students in their first year of a BSc in Information Technology, to aid understanding of computer programming. Research suggests that computer programming is a difficult subject both nationally and internationally and is responsible for high attrition rates, especially for students in their first year of a computer science course. This solution is based upon the researcher’s experience of being a student and also what the literature reveals. The proposed tool includes the use of a Raspberry Pi, which in itself was originally created for the purpose of education, and the python programming language, which is quoted as being a human readable language versus what is considered a more difficult programming language, namely Java.

The research outcomes from this study suggest that the tool needs some further consideration, especially the use of Python as the programming language, as the researcher found that the students were having difficulty understanding Java and the introduction of Python may cause additional confusion. Overall the research study and implementation approach proposed was considered a success due to the interest received from empirical research conducted.

The following attributes were successfully delivered:

* Research into student retention in Information Technology.
* Research into the available technologies.
* Manual for assembly of Raspberry Pi and setting up an operating system.
* Videos guiding students through stages of programming of the proposed game, namely Brick Collider:
  + Creation of the game window and escaping events.
  + Creation of Ball object and movement.
  + Creation of Paddle object and interactions with Ball object.
  + Creation of Bricks.
  + Creation of Brick and Ball interactions and Levels.
  + Adding Text for intro and levels.
  + Add game rules for losing lives.
* Empirical research through a focus group with semi-structured questions.

The purpose of this project is to engage students and further enhance their understanding of programming in an interesting way with the creation of a game. The tangible aspects are to show students the possibilities with programming. Further research is needed to bring the right product with the right results, especially in the choice of a language. It is the conclusion of the researcher that the choice of engagement programming language should match the language taught on the Information Technology course of the third level institution. It is also hard to conclude the effectiveness of such a learning engagement tool without implementation. To implement the tool would need a fresh year of students entering the institution and is beyond the timeframe of this project.

1. **Focus of Study**

Whilst in my first year of college in the College of Computing Technology, I saw a large number of students drop out. I was actually surprised at the number of students that didn’t make it to semester 2 and saw some of the problems with which some of my fellow students were having. There was one student I became friendly with, he was well versed in building desktop PC’s for gaming and was passionate about it, as he considered himself very good with computers. He however, was struggling with a couple of modules, Maths and in particular, Programming. He found it hard to associate maths and computing, but he was able to keep up. Programming, however was another matter entirely, he just could not grasp it and became very disconcerted with the course as a whole because of it. He was one of the few that I noticed that during that first semester were having difficulties and programming was a forerunner. I have to admit, I myself found it hard at the time as I could not imagine in my human way what the program was doing and for what purpose, I really had to sit down and spend a lot of time working it out.

These are the students that my project is aimed at, I chose this project to hopefully engage students that are on the verge of (and perhaps those that are even more so) dropping out of Information Technology courses at third level. It is my hope to develop a system that engages these students by blending real world technology (The Raspberry PI) and programming (Python) and give them the willingness to prevail and continue with their education in Information Technology. Students will get a hand’s on approach to building a system from components that are easily available and reasonably cheap to purchase, to choosing and installing an operation system, installing Python and some of its libraries and finally programming some simple games to give some real world experience to programming.

The end goal is to create a gaming console using easily available technology – The Raspberry Pi and recreate some classic and well loved games such as the likes of Brick Collider, Tetris, and perhaps Mini Golf. The Raspberry Pi will need an Operating System and an ability to run Python games. The Raspberry Pi can also be more than a Games Console it can function as any personal computer can and as such when started there should be an ability to go to games or another program such as a media player. Interfaces will need to be designed in accordance with standard practices to allow the user ease of use as with the more familiar devices they use. They will need to be easy to use and for all ages, stylistic and of a good resolution. There is also the possibility of opening the software up to be educational too, perhaps including some basic tutorials guiding children through school subjects of the Irish educational system.

The objective is to guide students through setup with the inclusion of a manual with pictorial representation and a video detailing each stage of the development of the Raspberry Pi as a games platform and help their understanding of subjects associated with third level. . To aid programming a video showing the development of a game is to be made for the game console, hopefully bring engagement through the joining of hardware and software.

1. **Literature Review**

Tinto (2006) states that the when the issue of student retention first arose approximately 40 years ago, the lack of Student retention was blamed upon the student, they were thought of as less able, less motivated, and less willing to reap the benefits that college graduation bestowed. Students failed, not institutions. This he states, is what we now refer to as blaming the victim. This view began to change in the 1970’s when role of the environment and that of the institution, was taken into account of the student’s decisions to stay or leave. Tinto states that this early work on student retention could be called “The age of involvement”. This showed that involvement mattered and that it most matters during the first year of college. Tinto (1997) reveals that classroom activities change the way that faculty and students interact in and out of the classroom and in so doing they influence persistence. Student engagement synonymous with involvement, (Carini, Kuh, & Klein, 2006) is one of the better predictors of learning and personal development. The more a student practices or studies a subject the more they will learn from it and the act of being engaged also adds to the foundation of skills and dispositions that is essential to live a productive and satisfying life after college.

It is clear that the need for engaging the students, especially in the first year of college is of great importance, as I have noticed in my first year this was indeed a hard time for a lot of students and attrition was high, nearly 50% of the class did not return for semester two from what I saw. I also remember looking at the class size in the second year and we were down to around 30 students from the original class in semester one of over 70 new students. The National Forum for the Enhancement of Teaching and Learning in Higher Education (2015) cited many causes for Student Non-completion of ICT courses, one of which was programming, based on the research of others. It is accepted that it is difficult to learn to program, high attrition rates are associated internationally with these programmes, especially in the early stages of college (Bennedsen & Caspersen, 2007). Attrition rates, for the first two semesters have been reported as 30 to 40% in many institutions and can be as high as 50% in the first semester, these figures are rapidly becoming the norm, in the early years of college (Beaubouef and Mason 2005). Learning to program does take practice, a lot of practice and I have experienced this first hand in my first semester, where I personally had to commit much time, patience and practice to fully understand what the program was doing and why it was doing what it was. Crenshaw et al (2008) say that large classes, limited computer labs along with limited time for mentoring students to develop key programming skills lead to conditions that do not give sufficient time for students to practise. Walker (2004) who says his computer science program distinguishes itself from many other academic programs with its low faculty to student ratio, schedule labs for most of their programming language courses to **1)** provide close teacher-student contact**, 2)** give students hands on experience, with on hand expertise, **3)** develop programming skills and **4)** freedom to experiment with ideas. He also states that the application of lab experiments provide for practice and drill which results in positive educational outcomes.

Clearly based upon the outcomes of many research papers by many authors, engagement is not only just about retaining students, it’s also about actively engaging the students in practices that will solidify their lessons learned to form a better understanding of what they are actually learning and possibly why. To properly engage students we could use some or all of the Cognitive Domain of Bloom’s Taxonomy to ensure labs would have a desired effect i.e. aid the understanding of programming by building something with technology. Bloom’s Taxonomy has three domains, The Cognitive Domain (Knowledge Based), The Affective Domain (Heart-Based Feelings) and The Psychomotor Domain (Action-Based). Clark (1999) explains the Cognitive Domain as involving knowledge and development of intellectual skills. The six major categories, according to an updated and possibly more accurate, are:

**Remembering:** Recalling learned Information such as lectures, notes etc.

**Understanding:** Comprehension of instructions and problems.

**Applying:** Apply what is learned in new situations.

**Analyzing:** Examine and break information into parts.

**Evaluating:** Making judgments on ideas or materials used.

**Creating:** Create a new structure from different elements.

In creating a lab for student engagement, all six categories are of great importance, but for my project I shall be concentrating on the first three categories, Remembering, Understanding and Applying. That is not to say that Analyzing, Evaluation or Creating is of any less importance, they are very important especially for student development and self confidence, but they are outside the scope of my project, which is to engage students and ease the harder subject of programming (for some) with some actual technology and build a platform for games. They can of course be added to fully reflect Bloom’s Taxonomy and perhaps answer another cause of student non-completion of ICT courses as included by the National Forum for the Enhancement of Teaching and Learning in Higher Education (2015) which say Academic skills are also another source of student’s not completing college.

Engaging students not only is important to keep students attending college, by engaging them we can possibly make their journey into and through college easier. I would like to at least see what I can do to make that journey easier. My project would hopefully engage students so they can physically make something whilst using programming skills learned in lectures to program some simple games. The set up alone could also include other lessons and perhaps students may also gain some experiences not currently taught in the curriculum in the first or the second semester. Although the project is aimed mainly at first and second semester students, we have an opportunity to continuously engage students into and perhaps beyond their third and forth semesters. We can also further incorporate Bloom’s Taxonomy to include the final three categories, by getting the students analysing and evaluating what they have done and then propose or even create something else based on what they have learned. I can see that there is a problem with student retention, I cannot say for a certainty that I have the answer, but this is what I propose I can do to help the situation as I see it. I hope it will indeed make a difference and change the minds of those considering that they cannot continue, those that feel programming is too hard and cannot see the value or what programming is used for and how it works.

Concerns about the decline in numbers and the skills levels of students wishing to enter computer science courses led to the idea of the Raspberry Pi. In 2006, Eben Upton, Rob Mullins, Jack Lang and Alan Mycroft, based at the University of Cambridge’s Computer Laboratory felt the need to do something about the situation. They identified a number of problems ICT curriculum was more focused on Microsoft’s Word and Excel, the end of the dot com boom and the rise of the home PC and games consoles to replace the ZX Spectrum, AMIGA Commodore 64 etc. There was not a lot they could do about an inadequate curriculum or the end of a financial bubble, but they could address the issues around computers which had become so expensive that experimentation had become forbidden by parents.

*“We don’t claim to have all the answers. We don’t think that the Raspberry Pi is a fix to all of the world’s computing issues; we do believe that we can be a catalyst. We want to see affordable, programmable computers everywhere. We want to break the paradigm where without spending hundreds of pounds on a PC, families can’t use the internet. We want owning a truly personal computer to be normal for children, and we’re looking forward to what the future has in store.”*(Raspberry Pi Foundation, 2015)

Stanford (2011) interviewed Alex Hope, the managing director and founder of Double Negative; he says that there is a shortage of people with computer programming skills and that it limiting growth. He would like to see more engagement of younger people with the excitement and potential of computer programming. One hopeful sign is the development of the Raspberry Pi, a computer that its makers intend it to be simple and cheap enough to break and fix just like his BBC Micro in the eighties. The Raspberry Pi Foundation was founded in 2008 to make the Raspberry Pi a reality and in February 2015 the sales of Raspberry Pi’s had reached five million (Upton, 2015). The Raspberry Pi was created to help increase the skills of precollege students, to get them involved with programming and would in my estimation be a great tool to use even in college for engagement. It can be utilised in many ways and there is a vast amount of material and projects which are openly available to all.

Beaubouef and Mason (2005) argue a point of what type of language and whether to teach objects early or later when teaching programming. They say McConnell and Burhans studied introductory text books for computer science and noted that the older textbooks (those on procedural languages) were on average less than 500 pages compared to newer textbooks such as Java which was 866 pages. They make no conclusions as to whether what language is better suited or if teaching objects early is a solution but they do say that it is a controversial topic and is not likely to go away anytime soon. Goldwasser and Letscher (2008) state that Pythons use, in academia has grown in recent years. Its attraction stems from its clean and simple syntax which allows students to give more when developing applications and learning. They use Python to introduce Computer Science students to programming and outline the differences between Python and Java. They say that it is a great first step in programming and later they then introduce Java and C++ to the module. In their conclusion they say that they are very pleased with the use of python finding that it gives a clear, coherent and consistent view of object-orientated programming. Python according to Python.org (2015) python is a high level language with dynamic semantics. It is an object orientated and interpreted language which has easy to learn syntax which emphasises readability. Prechelt (2000) empirically studied some programming languages and concluded that, amongst other findings designing programs in python take half the time of java and C or C++ and the resulting program is about half as long. The Raspberry Pi Operating System, Raspbian contains both Python and Java for programming but to engage student’s, I feel that Python would best suit as it is interpreted so there’s no need to compile the code, just run it and errors are caught at runtime. Also there is a huge amount of reference material online and videos to guide students further in other aspects of python programming with pygame. As python is also object orientated it is an aspect that can be brought in after some basic tutorials. Programs can be run procedurally in Python without the development of objects. An important step when introducing object orientation is the provision of tangible objects to interact with (Goldwasser and Letscher 2008). The games in my project all have objects and are tangible and as such would also possibly prove invaluable in the understanding of object-orientated programming.

In creating this project I will use a waterfall approach. Tutorialspoint.com (2016) state that the waterfall method was the first process model to be introduced, it is simple to use and understand. Each phase must be completed before the next can begin. It is suited to short projects where the requirements are well documented; it is not suited to projects which are likely to change. Hughey (2009) describes the waterfall approach as comprising of five phases:

**Requirements** phase – gathering detailed understanding of the user’s requirements.

**Design** phase – can be subdivided into a logical and a physical sub phase, where the system is designed (logically) without hardware or software considerations and then designed (physically) to hardware and software technologies specifications.

**Implementation** phase – this is where the code is written to the design.

**Verification** phase – ensure the project is meeting the expectations of the customer.

**Maintenance** phase – problems discovered are corrected during this phase.

Choosing the Waterfall methodology for the project fits in well with this project as the stages of development of this project are clear and documented I have only one requirement – the engagement of students in the first semesters of their course. This project does not hold all the answers to engagement it is my answer to a problem – the design is my own answer to the requirements that I have set and I shall implement the best solution as I figure it to be. Verification of the project shall be in the form of a presentation and group questions to students in their second semester at CCT of an Information Technology course, to ascertain if the project is well received and based on those findings maintenance can be administered.

If the project is ever introduced as a tool for engagement, there will be a need to produce software with a focus on change, this project needs the ability to change to meet the future needs of incoming students and as such, an Agile approach to software development would best suit the task of engaging the students, they after all, are the subject of the project and in essence the customers of the overall project. Agile techniques vary in practices they do share common characteristics, such as iterative development, interaction and the reduction of resource-intensive intermediate artefacts. Iterative development allows adaptation to changing requirements and working close together with good communication means decisions can be acted upon immediately. Reducing intermediate artefacts which add no value to the final deliverable means more resources can be devoted to development (Cohen, et al., 2003). In 2001 17 developers met to discuss lightweight development models their discussions brought forth the Manifesto for Agile Software Development. Beck, et al (2001) state, we are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

**Individuals and interactions** over processes and tools.

**Working software** over comprehensive documentation.

**Customer collaboration** over contract negotiation.

**Responding to change** over following a plan.

“That is, while there is value in the items on the right, we value the items on the left more”.

Agile techniques (or methods) include most notably, Extreme Programming (XP), Scrum and Feature Driven Development.

Cohen, et al (2003), states that Extreme Programming is the hottest methodology to emerge in recent years. Wells (2013) states that Extreme Programming improves a software project in five essential ways, communication, simplicity, feedback, respect, and courage. Extreme Programmers constantly communicate with their customers and fellow programmers. Kent (2000) gives us twelve rules for Extreme Programming:

**The Planning Game:** At the start of each iteration managers, developers and customers meet to prioritise the next release.

**Small Releases:** An initial version is put into production after first couple of iterations.

**Metaphor:** A set of metaphors are constructed by managers, developers and customers to develop the system.

**Simple Design:** Developers urged to keep it simple.

**Test:** Write tests first.

**Refactoring:**  Design should evolve to keep it as simple as possible.

**Pair Programming:** Two developers write code at the same machine.

**Continuous Integration:** Integrate new code as often as possible – test.

**Collective Ownership:** All code is owned by all developers and the can change it whenever they want.

**On-site Customer:** Customer should work with development team to answer questions.

**40 hour weeks:** Limit the need for overtime.

**Open workspace:** Developers work in a common workspace.

According to the Scrum Alliance (2015) Scrum is an Agile framework for completing complex projects. They surmise the Scrum framework in 30 seconds as, the product owner creates a prioritised list called the product backlog. They then plan a sprint, where the team pick tasks from the backlog and decide how to implement them. The team has a certain amount of time to complete (two to four weeks) – sprint, they hold daily scrums to assess progress. The Scrum Master keeps the team focused and at the end of the sprint the product should be potentially shippable. When the sprint is over there is a review. Cohen, et al (2003) state that Schwaber surmises the key points of scrum as:

Small working teams to maximise communication, minimise overhead and share knowledge.

Adaptability to changes to ensure the best possible product.

Frequent builds that can be tested, adjusted, documented and built upon.

Partitioning of work into clean low partitions or packets.

Constant testing and documentation as the product is built.

Ability to declare a product as done when needed.

Implementation of this project could be done using a form of the Scrum Agile Methodology; the students that I hope to engage are the customers, but in this case they can also be the Scrum team and the lecturer the Scrum Master. The engagement of the students cannot be static. It must have the ability to change to suit the needs of the students and as such it should have the ability to change rather than follow a strict plan. The end goal is to have working software to engage students although I have to create documentation with respect to building the project, the engaged students are free from that obligation. They are free to build, learn and engage in the project. The students themselves can be involved in the end goals of the project and as such there is a need for the ability to adapt to suit their needs, this is what Agile Software Development can give us. Using the Scrum Agile methodology will allow change and through various stages (sprints) we can declare a finished product.

1. **Research Design and Approach**
   1. **Research Questions**
   2. Are there any reasons why some students do not continue to the second semester and beyond?
   3. What can I do to help engage students in programming? Is there a technology that could help that engagement?
   4. What do current students think of an engagement project using Technology and programming?
   5. **Research Methodology**

Jacobsen (2013) defines Positivism as a philosophical positioning that emphasises empirical data and scientific methods. A researcher he says can infer knowledge about the real world by observing it, the world is full of regularities that are detectable. Experiments and statistics are the best methods with Positivism, comparative methods and case studies are used for theory testing and building. Larrain (1979) “*one of the features of positivism is precisely its postulate that scientific knowledge is the paradigm of valid knowledge, a postulate that indeed is never proved nor intended to be proved.”*That’s to say that Positivism is about knowledge, the gathering of knowledge and as such it really validates itself. Trochim (2006) states that positivism is the rejection of metaphysics, its goal is simply to describe the phenomena that we experience. The purpose of science he states is simply to stick with what we can observe and measure.

Pragmatism is another research approach to consider. McDermid (2016) says that Pragmatism claims that an ideology or proposition is true if it works satisfactorily. The meaning of a proposition is to be found in the consequences of accepting it, and that unpractical ideas are rejected. Alzheimer Europe (2013) describe it as using the best method, best suited to the research problem and not getting tied up in philosophical debates about the best approach. Pragmatic researchers have the freedom to use any methods, they recognise that each method has its limitations and different approaches can be complimentary. Creswell (2003) finds that pragmatism provides a basis for the following knowledge claims:

* It is not committed to any one system.
* Individual researchers have a freedom of choice – they decide on which procedures, techniques or methods are best for their research.
* Researchers look to the “what” and “how” to research based upon how it’s supposed to behave.

This project is partly about gathering knowledge but in the fact that it’s trying to answer a real world problem – Student not completing college courses, and in trying to answer that problem I must use a research approach that is reflective of my project. My approach to research for this project is a Pragmatism Approach, where I as the researcher can accept many different viewpoints as to why the problem exists and offer a solution.

* 1. **Research Methods**

I shall use a mixed methods approach gathering qualitative data through two primary methods, those being observation and semi-structured questions set into a presentation of my concept. This data will be useful in drawing conclusions specifically relating to my research questions. When the project has been completed, I would like to observe a group of first year students who have experienced firsthand the problems associated with programming by presenting my concept to them and posing a few semi-structured questions to them. The target group consisted of a sample size of 40, and represented the smaller of the two 1st year groups available for this research. The participants were students in their second semester of their first year of a BSc in IT programme studying at the College of Computing Technology. Not knowing these students I was looking for collective feedback to validate the importance of my concept. The use of this focus group was to gain a collective thought on the positives and/or negatives of my proposed solution, by means of:

Focus Group Observation – What do the student’s think about the proposed use of a new technology? Are there any difficult areas for them? How would be best to aid the students?

Focus Group Questions – What do the students think? Is the solution effective? Did they think they will find any areas difficult? Would it help them understand programming? Do they know about the Raspberry Pi?

* 1. **Ethical Review**

With this project there are ethical considerations which could potentially have a negative effect on the project as a whole. Students should be assured about anonymity and have given consent voluntarily, students should they wish, need not take part in evaluative activity. They should not be coerced, stressed or put under any pressure (Whitton, 2010). Would the project itself push the students who are considering dropping out further down that route? This project is not about class sizes or how lecturers engage with their students, it in its simplest form is my opportunity to consider what has happened to students through my observations and what I can do to help them with what is considered, through research, as the more challenging subjects, and through my own sensory experience of being a 1st year student within the same institution. It is about engaging students with technology and programming. It is about helping the understanding of programming and its value in Computer Sciences and Information Technology. The students who take part in the presentation will need assurances that the data collected will not be used for any other purposes than to evaluate the project. They also need to know that the data will be stored securely and will be destroyed when it is of no further use to this project. They are not individually identifiable and no student details have been recorded. There are generally a number of methods researchers employ to justify good research quality (Cresswell, 2003; Nolen & Van der Putten, 2007), these can be describe as follows:

1. The research must be ethical, for all researchers.
2. Research will not use language or words that are biased against persons because of gender, sexual orientation, racial or ethnic group, disability or age.
3. The research will involve the potential of suppressing, falsifying, or inventing findings to meet a researcher’s or an audience’s needs; these must be denied.
4. In planning a study, it is important to anticipate the repercussions of conducting the research on certain audiences and not to misuse the results to advantage one group.
5. An important issue in writing a scholarly manuscript is not to exploit the labour of colleagues and to provide authorship to individuals who substantially contribute to publication.
6. It is important to release details of the research with the study design so that readers can determine for themselves the credibility of the study.
7. **Data Collection & Analysis**

The collection of data will be gained from questions set into the presentation to the focus group. This gathered qualitative data which I interpreted as follows:

Holding the Raspberry Pi up – Does anyone know what this is?

*One student knew that I was showing the Raspberry Pi. She knew what it was but not a lot about it.*

There seemed to be general interest about the Raspberry Pi as it was handed around. Questions asked like what Operating System does it use, what is it for and what can it do? I answered all questions and tried to give as much information about the origins and the reason for its concept. Explaining that the Raspberry Pi can run on many operating systems and that swapping the SD card offers a lot of freedom and experimentation especially with OS’s

When asked, how do they feel about programming?

*One student ventured a groan and said terrible.*

I asked if there was any particular difficult area.

*The majority agreed it was all difficult.*

As I explained the struggles I had and the struggles I noticed of my fellow students in our first semester there was nodding in agreement of the difficulties these students faced were similar to that of my own. They too, to some extent agreed that trying to figure out what the program in fact was doing was their experience. The students, in my opinion were interested in the project and some were keen to view the videos I created and published on YouTube. They were interested in my experience of first year and how I personally got through the programming syllabus. Was the use of external tutorials useful? Would just following the course materials help? Was there any external websites which would provide useful tips or where were the best tutorials? I explained that there was a lot of information out there; the best of those would be the likes of Code Academy and Tutorials Point among others like Stack Overflow, W3Schools and YouTube. I also pointed out that the students themselves were a great resource as they themselves can help each other out.

When the presentation was concluded there was a coder dojo like session where students who were having certain difficulties were free to pick my brain. Prior to the presentation I ran through the sample code, for the session, to make myself more familiar with what could be asked about the code. The sample java code but was pushed to the side as the students sought answers to their own programming dilemmas with their main assignment and other lab work. I offered advice as to how I would have approached the building of the assignment. This was quite tough as I was not prepared for, nor had any prior knowledge of their assignment. I also noticed that not having practised Java in a while had its fall backs. I have become quite familiar with programming in Python and as such found myself trying to remember the lessons I had learned through my early years of college.

1. **Lessons Learned**

In answering my first research question, are there any reasons why some students do not continue to the second semester and beyond? I have learned that, there are student retention issues affecting third level institutes, it is well documented as are the most likely causes for Information Technology and Computer Science courses dropout rates of 30% – 40% (Beaubouef and Mason 2005). Perceived difficult subjects such as programming can often be attributed to at least part of the cause of early dropout rates for students in their first year of college. My opinion of my first semester is a good one, I feel that the faculty staff, at least the ones I had personal dealings with have on all occasions engaged with the students and have donated their spare time to engage further. To me, the main problem lies within the student’s difficulty with programming and is what is within the scope for which I can do something about.

Answering my first question led me to my second research question which has two parts. Having experienced firsthand the difficulties associated with programming, I was in a prime position to answer the first part, what can I do to help engage students in programming? I took what I felt as my obstacles and those that I perceived of my fellow students. What is this program doing and why does it do it? I don’t see why it’s needed. I decided to create something tangible that we could make with programming – a game. In creating the game, I could bring in the lessons learned in the first two semesters of a programming curriculum. If statements, for loops, while loops and aspects of Object Orienting are some of the lessons I hoped to explain. While researching some of the programming problems I faced throughout my previous three years of college, I have come across many tutorials on websites and YouTube; this led me to the idea of creating video tutorials where one can explain step by step the choices made and the functionality caused by the chosen code. To take tangibility a step further and actually have a physical object that would bridge the gap between technology and programming led to the second part of my second question, is there a technology that could help that engagement? There indeed was, the Raspberry Pi. This wonderful piece of technology was created with teaching programming amongst other things, in mind. Since the creation of this project seen a new Raspberry Pi go on release, the even more powerful Raspberry Pi 3.

My third and final question, what do current students think of an engagement project using Technology and programming, is where I learned most about my project. To keep students engaged, I assumed that we need to keep them interested in the subject that they had difficulty with – programming. That I feel is still the case, having presented the students in their second semester of an Information Technology course at CCT, I personally got to see how they reacted to my project. They were indeed interested and vocal about the project and its proposals. Some had, even within the first 24 hours watched the videos I published on YouTube. But having had a chance after the presentation to deal with them on a more one to one, it transpired that they were having problems with their main assignments and wanted to know more about how I would deal with the problems they were having. I presumed that Python was indeed a great language especially for engaging students to help them understand programming, and it is but, why put struggling students through another language? I am now of the opinion that the true engagement would have to be in the language that they are currently learning. Sure Python is great for developing Object Orienting skills and what can be written in Java can be written easier or more human readable form, but it is still another language, another set of rules and another way of writing for students to have to grasp. I love to program but having four years of practice at it and three years more than the current first years at CCT, I may have truly forgotten how tough it was and having reached the stage I’m at, I’m sure that I now find it easier to jump from language to language than I would have in my first year.

Although I gained some great insight into the importance of engagement and the methods behind lecturing with the use of Blooms Taxonomy, I feel that the year of building the project needs more to validate my concept for student engagement. The insights from my observations of the focus group of first year students were certainly positive, but they due to time constraints didn’t have the opportunity to build the game or game console. The full implementation of the project to a fresh batch of students would yield more insight as it also holds the ability for the next generation of first years to actually get their hands dirty and actually build the game and game console. It would be then I would see the true results of my concept, for better or worse.

As stated this project is not intended to change the curriculum of the college or even add to it, it is my concept to engage and aid understanding of programming and in doing so perhaps an offshoot could be a better understanding of what the students perceive as difficult with their understanding with programming. If the concept is ever introduced, it may point to areas in the curriculum where most difficulties are found. Future students may benefit from the previous year’s learning’s and this engagement concept could adapt to suit those needs.

In creating the videos, I came across some of my failings, speed of typing with programming and a difficulty to enunciate. I knew what I wanted to program and how to do it but combining it with an explanation of what the program was to do with the code and why I chose that code caught up with my typing skills and ability to properly express myself. Many takes were needed to complete each video and proved to be very time consuming. The more videos were made, I found myself getting more comfortable with my ability to express what I wanted. I do, however still feel that my videos could be all redone and made better, but where does one stop? These videos are supplementary to the creation of the game and the true engagement would come from the creation of the game and console whilst surrounded by students and where they are free to ask questions and build.

Shortcuts do not always work out. In creating a program it may appear quicker to copy and paste sections of code that similar or the same, make sure to change the differences no matter how subtle they are. When changing the *block\_collide* class I copied and pasted two lines of code into three if statements one should have changed the *dirx* value but because it was a copy it also changed the *diry* value and as a result when the ball hit the bricks on the sides it behaved like it hit them on the top or bottom. This cost me time and effort that I could have saved otherwise.

1. **Conclusions and Recommendations**

It is clear that programming is difficult and can be at least partially responsible for lack of retention. This project is my way to give those only starting in programming a little help with understanding how its features interact and the rules that govern it can be applied in a fun and engaging way. Most people I know have played or at least know of Breakout, so I decided that what better way to engage than creating it, explaining it step by step and using technology create a game console which can be replicated and passed around. This is only my first stem into creating games for engagement or explaining programming and it could be as big as I wish it to be. I can from here step further with Brick Collider by creating extra game features as mentioned in [**Appendix 4: Implementation of the system**](#Appendix_4), to make a fully working game that would have a better longevity than the current version, like scoring and the saving of High Scores which can be handled with external files and as such could show read/write of files in programming. It could also be left as is so that the student could if he/she desires create their own distribution of the game so that they could build on their creativity and perhaps engage themselves further with programming. Other games could be created also, like a Tetris clone, which could be used to explain *Arrays.* If I were to create a Tetris clone it would involve array blocks being set on the larger array of the screen. When one line (or row of the array) is full it could be demolished. Later semesters in the programming curriculum see GUI Programming this would be a great opportunity to expand on the project with making a game interface for the selection of the games. Using or expanding on the lessons learned during the lectures a student could further this project and his/her knowledge with programming.

I chose Python for the programming language for its ease of learning and almost human readability because I thought what better way to demonstrate the core concepts of programming than Python. Perhaps it is a great way to learn programming but that is not my mission here. Since meeting with the students I have come to realise that perhaps my approach was flawed and that perhaps learning to program in a different language would be better suited to later stages in college when there is a greater understanding of the whole programming core. Why push a new language when there is clearly a full understanding with the language that is currently being mastered? Should this project ever be utilised (in CCT or elsewhere), I would recommend the use of the current language being taught, in the case of CCT that would be Java.

Even at an early stage, perhaps ever so briefly, the final three stages of Blooms Taxonomy could be introduced. Having the students Analyse, Evaluate and Create could further expand on the understanding of programming. Why was the *if* statement used why not a switch? What would happen if I used a switch? Giving reasons for choices and go create something new based on some of the aspects included here. This could be included in more general terms like a debate or a Q & A session, rather than the production of documentation.

I find it fulfilling to solve those little or large problems associated with programming, hours sometimes spent trying to master what probably, is only a little problem and the final relief when the problem is found and solved. It is how I learn, by looking for and finding solutions to the problems I’ve encountered or solving them without aid, overall this project is about providing examples to ease transition into programming. In it I also hope to have answered some of, if not all the problems I have found when I first started learning programming.

1. **Reference List**

Alzheimer Europe (2013). The four main approaches. Available at: <http://www.alzheimer-europe.org/Research/Understanding-dementia-research/Types-of-research/The-four-main-approaches> [Accessed 02 February 2016]

Beaubouef, T. and Mason, J., (2005) Why the high attrition rate for computer science students: some thoughts and observations, SIGCSE Bull., 37:2,103-106.

Beck, K., (2000) *Extreme Programming Explained: Embracing Change*, Addison-Wesley.

Beck, K. et al. (2001). *Manifesto for Agile Software Development.* Available at: <http://agilemanifesto.org/> [Accessed 19 January 2016].

Bennedsen, J. and Caspersen, M.E., (2007) *Failure rates in introductory programming*, SIGCSE Bull., 39:2, 32-36.

Carini, R. M., Kuh, G. D., & Klein, S. P. (2006). *Student engagement and student learning: Testing the linkage. Research in Higher Education*, 47(1), 1-32.

Clark, D. (1999), Learning Domains of Bloom's Taxonomy. Available at: <http://www.nwlink.com/~donclark/hrd/bloom.html>. [Accessed 13 December 2015].

Cohen, D., Lindvall, M., Costa, P. (2003), DACS State-of-the Art/Practice Report Agile Software Development. Available at: <https://www.google.ie/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&ved=0ahUKEwiIhtqng_3LAhXDiA8KHaTOA2oQFggqMAE&url=http%3A%2F%2Fciteseerx.ist.psu.edu%2Fviewdoc%2Fdownload%3Fdoi%3D10.1.1.201.2704%26rep%3Drep1%26type%3Dpdf&usg=AFQjCNEfLATseionszkWorqbcn-g40IEiA&sig2=qF6f4KqWqVb22QEAkKUmZg&cad=rja> [Accessed 18 January 2016].

Crenshaw, T.L., Chambers, E.W., Metcalf, H., and Thakkar, U., (2008) A case study of retention practices at the University of Illinois at Urbana-Champaign, SIGCSE Bull. 40:1, 412-416. Available at: <http://mathcs.slu.edu/~chambers/papers/sigcse235.pdf>

Creswell, J. (2003). *Research design: Qualitative, quantitative, and mixed methods approaches.* (2nd ed.) Thousand Oaks: Sage.

Goldwasser, M., Letscher, D. (2008) Teaching an Object-Oriented CS1 — with Python SIGCSE Bull. 40:3, 42-46. Available at: <http://dx.doi.org/10.1145/1597849.1384285>

Henry, J., Henry, S., (1993) *Quantitative assessment of the software maintenance process and requirements volatility*. In Proc. of the ACM Conference on Computer Science, pp 346–351.

Hughey, D. (2009) *The Traditional Waterfall Approach* Available at: <http://www.umsl.edu/~hugheyd/is6840/waterfall.html> [Accessed 1 April 2016].

Jakobsen, T. (2013) *Theory of Science – What is Positivism?* Available at: <http://www.popularsocialscience.com/2013/02/15/theory-of-science-what-is-positivism/> [Accessed 01 February 2016].

Langer, E. (1997) *The power of mindful learning*. Reading, MA: Addison Wesley.

Larrain, J. (1979). *The Concept of Ideology.* London: Hutchinson*.* pp 197.

Mackenzie, N., Knipe, S., (2006) Research *Dilemmas: Paradigms, methods and methodology.* Issues In Educational Research, Vol 16, 2006.

McDermid, D. (2016) Pragmatism*.* Available at: <http://www.iep.utm.edu/pragmati/> [Accessed 01 February 2016].

Nolen, A. L., & Van der Putten, J. (2007). *Action Research in Education: Addressing Gaps in Ethical Principles and Practices*.  Educational Researcher, 36(7), 401-407.

Prechelt, L. (2000) An empirical comparison of C, C++, Java, Perl, Python, Rexx, and Tcl Available at: <http://page.mi.fu-berlin.de/prechelt/Biblio/jccpprt_computer2000.pdf> [Accessed 15 February 2016].

Python.org (2016) *What is Python? Executive Summary.* Available at: <https://www.python.org/doc/essays/blurb/> [Accessed 15 February 2016].

Raspberry Pi Foundation (2015) About Us Available at: <https://www.raspberrypi.org/about/> [Accessed 26 January 2016].

Schreiner, L., Louis, M., (2000) *Measuring Engaged Learning in College Students: Beyond the Borders of NSSE.*

Scrum Alliance (2015) *Learn About Scrum* Available at: <https://www.scrumalliance.org/why-scrum> [Accessed 19 January 2016].

Stanford, P 2011 ‘Computing classes don’t teach programming skills’, The Telegraph 03 December, Available at: <http://www.telegraph.co.uk/technology/video-games/8931387/Computing-classes-dont-teach-programming-skills.html> [Accessed 28 January 2016].

Tinto, V., (2006) *Research and practice of student retention: what next?* Available at: <https://www.uaa.alaska.edu/governance/facultysenate/upload/JCSR_Tinto_2006-07_Retention.pdf> [Accessed 09 December 2015].

Tinto, V., (1997) *Colleges as communities: Exploring the educational character of student persistence.* Journal of Higher Education, 68, pp. 599-623.

Trochim, W. (2006) *Positivism & Post-Positivism.* Available at: <http://www.socialresearchmethods.net/kb/positvsm.php> [Accessed 01 February 2016].

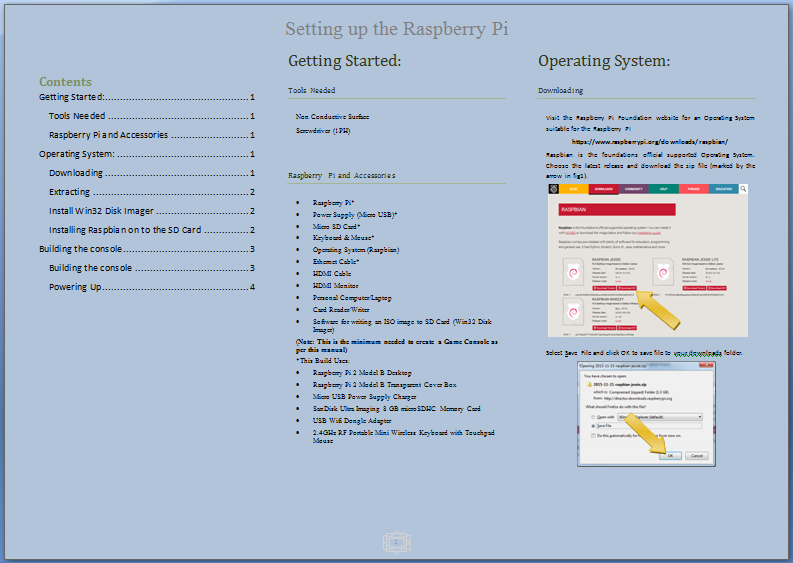
Tutorialspoint.com (2016) *SDLC – Waterfall Model* Available at: <http://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm> [Accessed 1 April 2016].

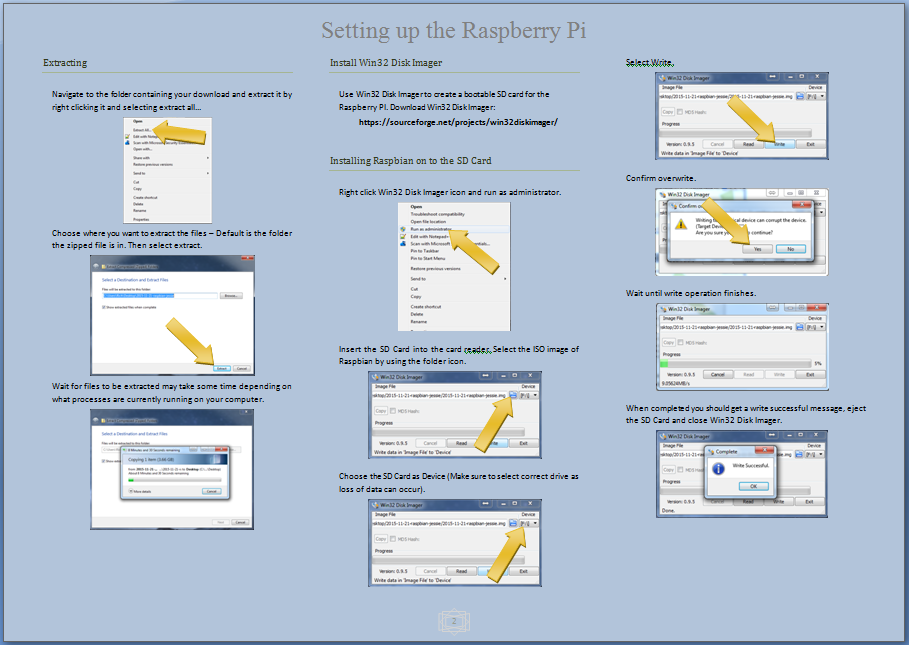
Upton, L (2015) *FIVE MILLION SOLD!* Available at: <https://www.raspberrypi.org/blog/five-million-sold/> [Accessed 28 January 2016].

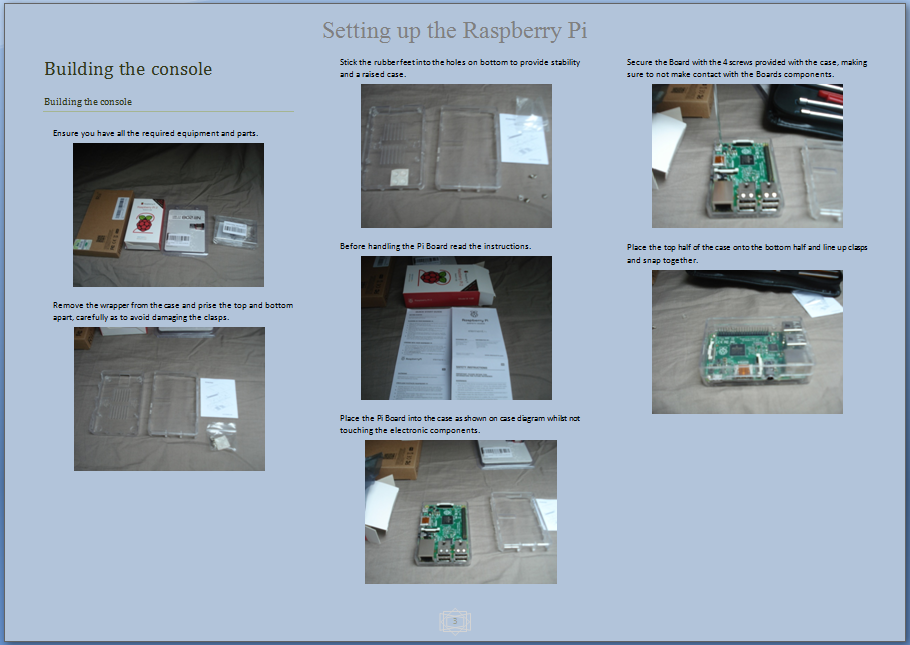
Walker, G. (2004) *Experimentation in the Computer Programming Lab.* Available at: <http://dx.doi.org/10.1145/1041624.1041660> [Accessed 12 December 2015].

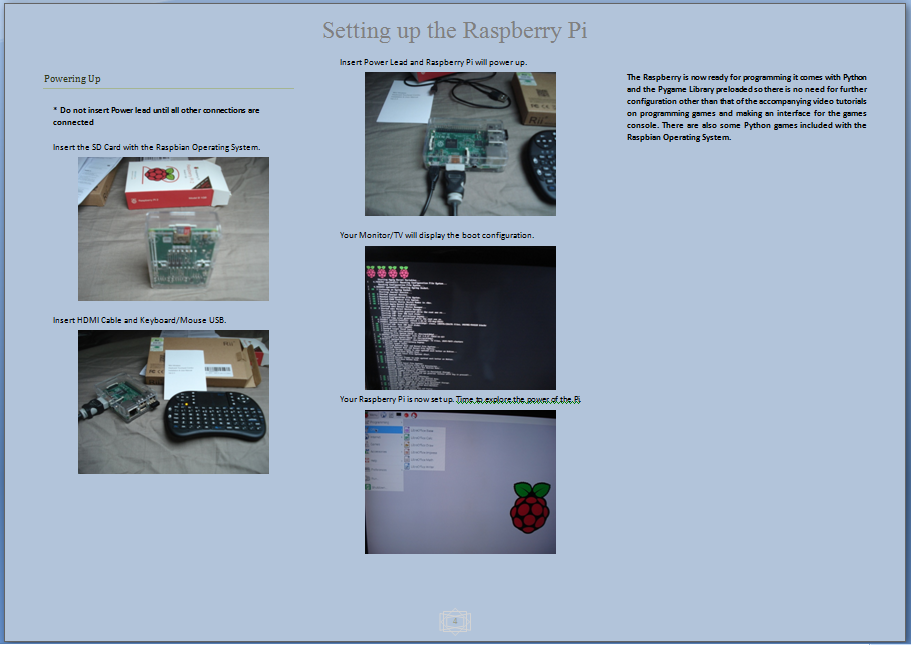
Whitton, N. (2010) *Learning with Digital Games A Practical Guide to Engaging Students in Higher Education.* New York: Routledge.

1. **Appendices**
   1. **Appendix 1: Manual**

****

****

****

****

* 1. **Appendix 2: Systems Analysis**

This project is about engaging students using easily obtainable and low costing technology, the Raspberry Pi with programming. It will guide students through the set up of an Operating System and on the Raspberry Pi, downloading of Python and some needed libraries for game functionality, the creation of some games and interfaces for menus. Students in their first semester and even in their second semester, as I have noticed and indeed as has the authors of literature about student engagement in the computer sciences, are having difficulties with programming. An Agile Software Development approach will be used for the deployment of the project, as it allows a quick response to the changing of the overall project. Scrum would be the most applicable methodology as the project itself can be broken down into sprints. After each sprint there is the potential to ship a viable product. The students would be the customers and also the scrum team and the project as a result would need to change based upon their needs. The creation of the project itself needs a different approach, the waterfall approach which has five phases closely related to how the project such as this is created.

The finished system will have an Operating System, a manual to complete start up and videos to run through programming of Brick Collider to at least a basic level for engagement and explanation of some basic terms used in programming with Python and Pygame. Further sprints can be developed to engage students with other semesters of college. New games can be added to the console through coding of the main interface. The deliverables for this project are:

**Hardware:**

* Raspberry Pi
* Start up manual/Video
* Programming Video for Engagement

**Software:**

* Operating System.
* Python and Pygame.
* Games:
  + Brick Collider
  + Tetris
  + Mini Golf (Time permitted).
* Interfaces (Time permitted).

**Aspirational:**

* Educational game.
* Add new game through interface.

**The Waterfall Approach**

The waterfall approach shall be utilised to create this project. The waterfall approach has five phases which coincide with this project.

**Requirements** phase – gathering detailed understanding of the user’s requirements. This is the information gathering of the problem. Has the engagement of students on IT/Computer Science courses been documented? Are there reasons for student dropping out of higher education on these courses? Are there any programs to engage students on these courses or any means to alleviate the difficulties found? This phase shall be covered in the Literary Review.

**Design** phase – can be subdivided into a logical and a physical sub phase, where the system is designed (logically) without hardware or software considerations and then designed (physically) to hardware and software technologies specifications. In the Logical sub phase I shall design my answers to the Requirements phase, what can be done to ease the problems of engaging students on IT/Computer Science courses with the subject they are finding hard – programming. In the physical sub phase I shall design a system from my logical design using hardware and software. Choices in the Design phase are covered in the Literary Review.

**Implementation** phase – this is where the code is written to the design. In the Implementation phase I shall use my design to implement my answer to student engagement using the hardware and software choices. This is build stage of the project and is covered in Appendix 4: Implementation of the system.

**Verification** phase – ensure the project is meeting the expectations of the customer. The verification phase shall consist of a short presentation of the project to students in their first year, followed by a questionnaire to gauge if they find the project interesting and if they have any suggestions for improvement.

**Maintenance** phase – problems discovered are corrected during this phase. The Maintenance phase is unfortunately outside the time scale of the project. This phase happens with full deployment of the project. Debugging any issues found, matching the project to the needs of engaging the students.

**Scrum**

The system will upon completion have a User Interface which will list the games with choices of Play or Description or add game to the menu. If description is selected it should open to an expanded Interface with a full description of the game and a title picture. The choices here are return to previous menu or play the game. Selecting Play should automatically start the game.

* **Sprint 1:**
  + Manual/video to guide students through the initial steps of setting up the Raspberry Pi.
  + Set up Raspberry Pi.
* **Sprint 2:**
  + Manual/video to guide students in setting up an Operating System on the Raspberry Pi.
  + Install Operating System, Python and needed libraries.
* **Sprint 3:**
  + Build Brick Collider.
  + Video for programming Brick Collider
* **Sprint 4:**
  + Build game 2.
* **Sprint 5:**
  + Build an interface.
  + Add games to interface.

**Use Case Diagram:**

Raspberry Pi

User

**User Case Descriptions**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | Rasp-01 | | | |
| **Use Case Name:** | Prepare for Operating System. | | | |
| **Created By:** | Richard Loy | | **Last Updated By:** |  |
| **Date Created:** | 21-01-16 | | **Last Revision Date:** |  |
| **Actors:** | | User | | |
| **Description:** | | User starts to prepare Raspberry Pi | | |
| **Trigger:** | | User wants to set up a Raspberry Pi. | | |
| **Preconditions:** | | User must have a Laptop, an ISO writer program and a Raspberry Pi compatible SD Card. Laptop must have Card slot or peripheral device for SD Card. | | |
| **Post-Conditions:** | | User will have an Operating System ready for Raspberry Pi. | | |
| **Normal Flow:** | | 1. User reads Manual. 2. User browses to download site. 3. User selects the OS to download. 4. User downloads the ISO Operating System File. 5. User inserts SD Card into SD Card slot. 6. User runs ISO writer and writes ISO to SD Card. 7. User ejects SD Card when writing is finished. | | |
| **Alternative Flows:** | | 1. User has ISO OS image 2. User inserts SD Card into SD Card slot. 3. User runs ISO writer and writes ISO to SD Card. 4. User ejects SD Card when writing is finished. | | |
| **Exceptions:** | | 2 No games exist. | | |
| **Includes:** | |  | | |
| **Frequency of Use:** | | 1 time. | | |
| **Special Requirements:** | |  | | |
| **Assumptions:** | | User has Internet connection. | | |
| **Notes and Issues:** | |  | | |

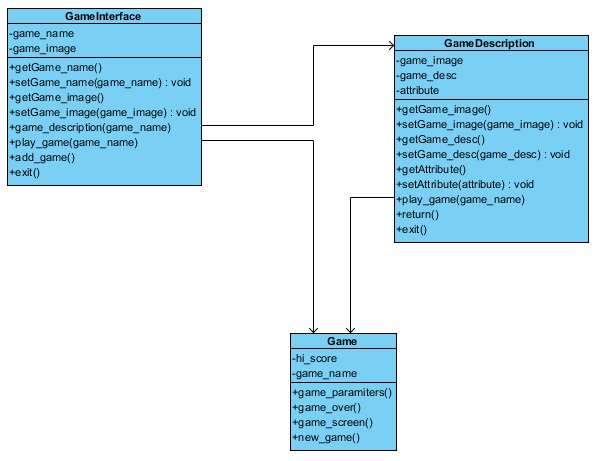
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | Rasp-02 | | | |
| **Use Case Name:** | Set up Raspberry Pi. | | | |
| **Created By:** | Richard Loy | | **Last Updated By:** |  |
| **Date Created:** | 21-01-16 | | **Last Revision Date:** |  |
| **Actors:** | | User | | |
| **Description:** | | User Sets up Raspberry Pi | | |
| **Trigger:** | | User wants Start up the Raspberry Pi for the first time. | | |
| **Preconditions:** | | User must have Raspberry Pi and peripherals | | |
| **Post-Conditions:** | | User will have a set up Raspberry Pi. | | |
| **Normal Flow:** | | 1. User inserts SD Card with OS 2. User inserts power lead into Raspberry PI. 3. User follows instructions in Manual for initial setup. | | |
| **Alternative Flows:** | |  | | |
| **Exceptions:** | | 1. No SD Card exists. | | |
| **Includes:** | |  | | |
| **Frequency of Use:** | | 1 time. | | |
| **Special Requirements:** | |  | | |
| **Assumptions:** | | User has Manual for reference. | | |
| **Notes and Issues:** | |  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | Rasp-03 | | | |
| **Use Case Name:** | User installs Python Libraries. | | | |
| **Created By:** | Richard Loy | | **Last Updated By:** |  |
| **Date Created:** | 21-01-16 | | **Last Revision Date:** |  |
| **Actors:** | | User | | |
| **Description:** | | User sets up the Raspberry Pi for Games. | | |
| **Trigger:** | | User wants to be able to play games on the Raspberry Pi. | | |
| **Preconditions:** | | Raspberry Pi must be powered on. User has Manual. | | |
| **Post-Conditions:** | | Raspberry Pi set up for gaming. | | |
| **Normal Flow:** | | 1. User reads manual. 2. User opens command line. 3. User enters command from manual. 4. The Libraries are downloaded. | | |
| **Alternative Flows:** | |  | | |
| **Exceptions:** | |  | | |
| **Includes:** | |  | | |
| **Frequency of Use:** | |  | | |
| **Special Requirements:** | |  | | |
| **Assumptions:** | | User has a Raspberry Pi set up with previous steps. | | |
| **Notes and Issues:** | |  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | Rasp-04 | | | |
| **Use Case Name:** | User selects a game. | | | |
| **Created By:** | Richard Loy | | **Last Updated By:** | RL |
| **Date Created:** | 21-01-16 | | **Last Revision Date:** | 23-01-16 |
| **Actors:** | | User | | |
| **Description:** | | User decides to play a game on the Raspberry Pi Games console. | | |
| **Trigger:** | | User wants to be entertained. | | |
| **Preconditions:** | | Raspberry Pi must be powered on. | | |
| **Post-Conditions:** | | User selected game to play. | | |
| **Normal Flow:** | | 1. User starts game interface. 2. User browses the games. 3. User selects a game. 4. User chooses to check description. 5. User selects play. 6. User plays the game. | | |
| **Alternative Flows:** | | 1. User starts game interface. 2. User browses the games. 3. User selects a game. 4. User selects play. 5. User plays the game. | | |
| **Exceptions:** | | 2 No games exist. | | |
| **Includes:** | |  | | |
| **Frequency of Use:** | |  | | |
| **Special Requirements:** | |  | | |
| **Assumptions:** | | User has a Raspberry Pi with games. | | |
| **Notes and Issues:** | |  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | Rasp-05 | | | |
| **Use Case Name:** | User adds a game (Aspirational Deliverable). | | | |
| **Created By:** | Richard Loy | | **Last Updated By:** |  |
| **Date Created:** | 21-01-16 | | **Last Revision Date:** |  |
| **Actors:** | | User. | | |
| **Description:** | | User decides to add a game to the Raspberry Pi Games console. | | |
| **Trigger:** | | User has created or obtained a new game. | | |
| **Preconditions:** | | Raspberry Pi must be powered on. | | |
| **Post-Conditions:** | | User added a game to play. | | |
| **Normal Flow:** | | 1. User navigates to game interface. 2. User selects add game. 3. User selects game to be added. 4. Game is automatically added. | | |
| **Alternative Flows:** | |  | | |
| **Exceptions:** | | 2 Game is not in correct directory. | | |
| **Includes:** | |  | | |
| **Frequency of Use:** | |  | | |
| **Special Requirements:** | |  | | |
| **Assumptions:** | | User has a Raspberry Pi with games interface. | | |
| **Notes and Issues:** | |  | | |

**High Level Class Diagram:**

****

* 1. **Appendix 3: Systems Design**

**User Interface Design**

**Main Game Console Interface**

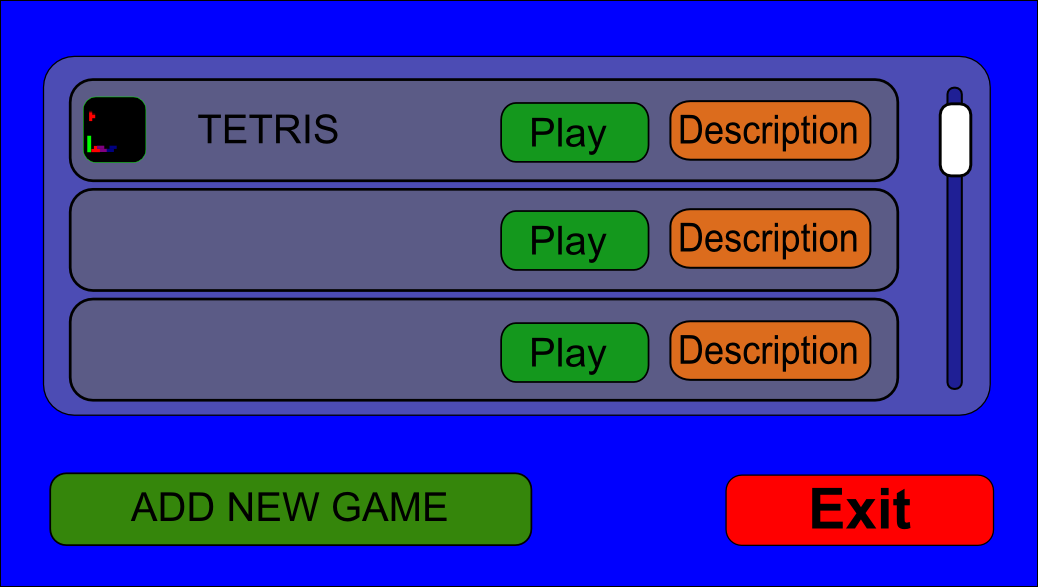
The user interface is the starting interface for the games console. This lists the games currently added to the console. From here, the user should be able to select games and play them directly or select a description to see a description of the game. There should be an option to end the console to allow the user to interact with other aspects of the Raspberry Pi.

Buttons:

* Play – should start the game directly and bring the user to the game screen for the options associated with the selected game.
* Description – the user should be brought to the Game Description Interface.
* Exit – close the games console.

Tooltips:

* Play – should display “Play the game”.
* Description – should display “See description for the game”.
* Exit – should display “Close Games Console”.



Main Game Console Interface

**Game Description Interface**

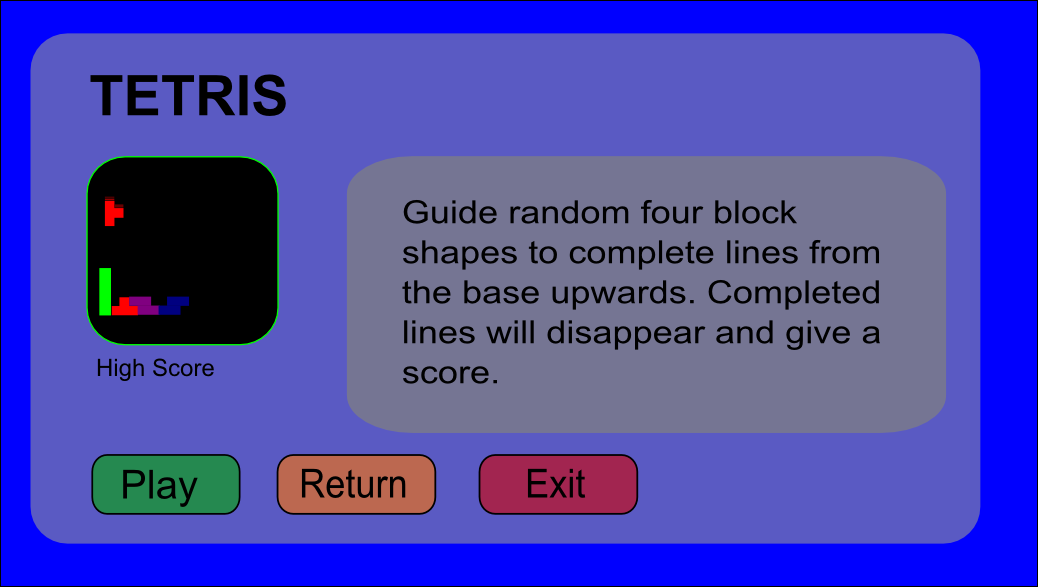
The Description Interface will have a large image of the game and a description of the game. Options are, play the game, return to Main Game Interface or exit the games console.

Buttons:

* Play – should start the game directly and bring the user to the game screen for the options associated with the selected game.
* Return – the user should be brought back to the Main Game Console Interface.
* Exit – close the games console.

Tooltips:

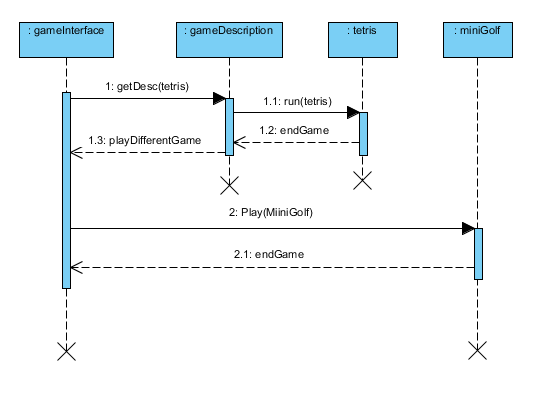
* Play – should display “Play the game”.
* Return – should display “Select a different game”.
* Exit – should display “Close Games Console”.



Game Description Interface

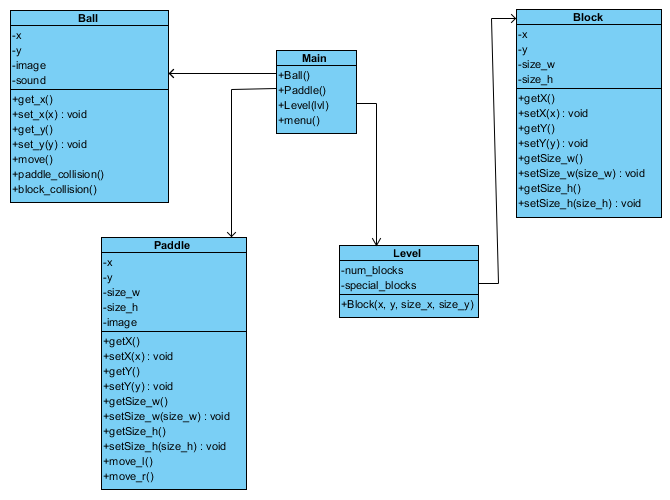
**Sequence Diagram**

This shows the sequence of events for the user selecting a game and getting the description of that game. From that interface the user chooses to play the game. When finished they then chose to select a different game and played it directly from the Main Game Interface. On completion the user chose to not play anymore and ended the sequence.

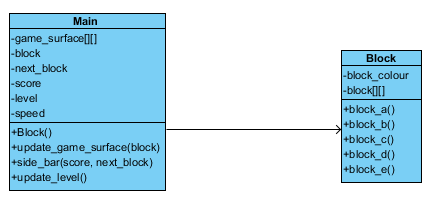


**Class Diagrams**

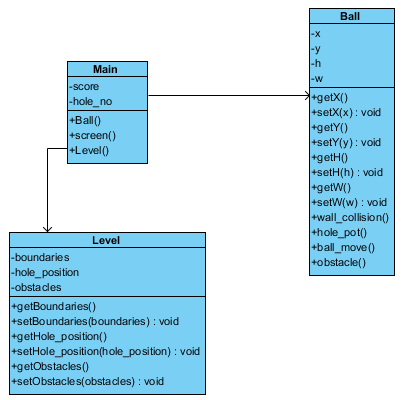
**Brick Collider**



**Tetris**

****

**Mini Golf**



* 1. **Appendix 4: Implementation of the system**

The construction of the project takes many stages from the initial purchase of the hardware to host the games console and the peripherals needed to operate the Raspberry Pi, construction of a manual and the filming of short videos to describe and make the tasks of building the Raspberry Pi Games Console easier, filming instructional videos to coincide with the coding of a game for engagement of Python Programming (Brick Collider) to the making of interfaces to display the games on the console.

**Hardware and Software Assembly**

Implementation started with the purchase of the needed hardware:

* [Raspberry Pi 2 Model B Desktop](http://www.amazon.co.uk/Raspberry-Pi-Model-Desktop-Linux/dp/B00T2U7R7I)
* [Raspberry Pi 2 Model B Transparent Cover Box](http://www.amazon.co.uk/Tontec-Raspberry-Model-Transparent-Enclosure/dp/B00NUN98UW/ref=dp_ob_title_ce)
* [Micro USB Power Supply Charger](https://www.amazon.co.uk/gp/product/B00AUKR4EU/ref=od_aui_detailpages00?ie=UTF8&psc=1)
* [SanDisk Ultra Imaging 8 GB microSDHC Memory Card](https://www.amazon.co.uk/gp/product/B00MBTKT6S/ref=od_aui_detailpages00?ie=UTF8&psc=1)
* [USB Wifi Dongle Adapter](https://www.amazon.co.uk/gp/product/B010AKMF3Y/ref=od_aui_detailpages00?ie=UTF8&psc=1)
* [2.4GHz RF Portable Mini Wireless Keyboard with Touchpad Mouse](https://www.amazon.co.uk/gp/product/B016A6NSL0/ref=od_aui_detailpages00?ie=UTF8&psc=1)

Total cost was €90 (inc. delivery). The wireless keyboard with touchpad mouse was chosen for convenience as it provides a mouse and a keyboard in one unit and only uses one USB port and the transparent case was chosen to protect the Raspberry Pi and give a more technical feel to the build as one can see the inner workings. All chosen items are compatible with the Raspberry Pi 2 Model B. To get the Raspberry Pi working fully we also need:

* HDMI Cable
* HDMI TV/Monitor
* Laptop
* Raspbian Operating System
* Card Reader (to transfer OS to SD Card)
* ISO file writing software (Win32 Disk Imager or similar)
* Software
* Python 3.4.4
* Pygame 1.9.2 (cp 3.4, win amd64 – for windows )
* Numpy 1.10.4 (cp 3.4, win amd64 – for windows)
* Tkinter

**Manual for Creating the Raspberry Console**

The manual was devised to aid the assembly of the Raspberry Pi; it is a pictorial representation and describes how to build the Raspberry Pi from downloading the Raspbian Operating System and installing it onto a SD Card to the assembly of all the components of the Raspberry Pi. Each stage in the creation and build of the Raspberry Pi was documented, taking numerous screen shots of the stages and photos of the assembly of the Raspberry Pi. The manual was designed using Microsoft Word, dividing a landscaped page into three columns and using simple English with as short sentences as possible to describe the actions performed. It has accompanying videos for those that would prefer to see a more real time assembly of the Raspberry Pi Games Console than that provided by the manual.

**Videos for Creating the Raspberry Console**

The videos were also devised to aid the assembly of the Raspberry Pi, from downloading the Raspbian Operating System and installing it on a SD Card through to the assembly of the Raspberry Pi. As with the manual each stage is captured either using a screen capture program (CamStudio) or using a camera. These are in real time only skipping the time consuming aspects of downloading the Raspbian Operating System, extracting it and writing it to the MicroSDHC Card. In creating the videos – which took many takes, I wrote a script for narration purposes but for the most part I adlibbed and used the script for guidance only. This was my first ever experience with filming myself and it is not as easy as it would seem. When completed the videos were uploaded to YouTube and set to public so everyone can view:

Raspberry Pi Games Console Part 1: Download and Write:

<https://www.youtube.com/watch?v=x6EoyfQ9_iY>

Raspberry Pi Games Console Part 2: Raspberry Pi Assembly:

<https://youtu.be/WfTYPTnb0zM>

**Games**

**Brick Collider**

This is the game I chose to use to engage the student’s with programming, it is a version of an old 70’s game Breakout (designed by Steve Jobs and Steve Wozniac in 1975 for Atari). I chose this game as my engagement game because it blends many aspects of programming from the start. ***While*** loops for game loop or pause loop, ***for*** loops to get events and brick creation, ***if*** statements to check conditions like ball position and events and it also incorporates ***inheritance*** and ***object orientated*** programming. It has four objects **1)** The Paddle, **2)** The Bricks, **3)** The Ball which interacts with both and finally **4)** The BrickCollider which has the main functions for game loop, level creation and text introductions. Seven video tutorials will guide the students through the stages of making a simplified version of Brick Collider. They are as follows:

**1)** Making a Pygame screen and ensuring it is centred on the screen

**2)** Create the Ball object and have it moving around the screen.

**3)** Create the Paddle object and create functions for the ball to bounce off the paddle if they collide.

**4)** Create Brick objects

**5)** Add collision destruction for Bricks and create Levels

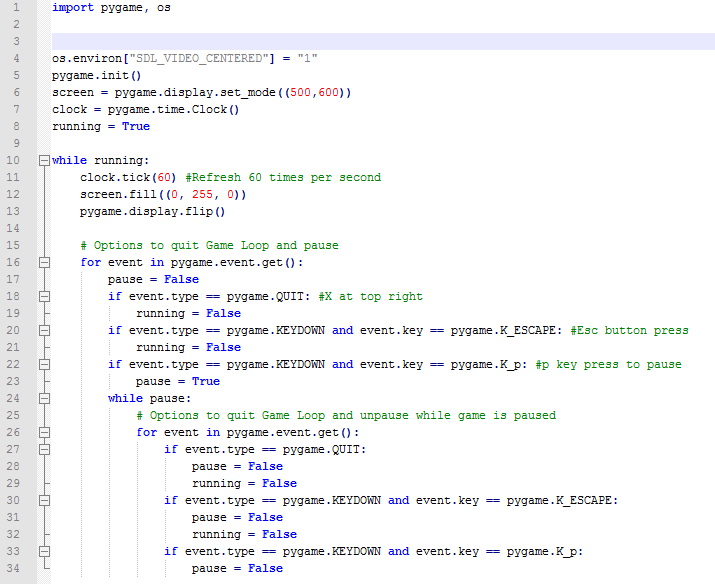
**6)** Add Text for Intro to game and count down for level

**7)** Add lives and lose lives and create a new game option.

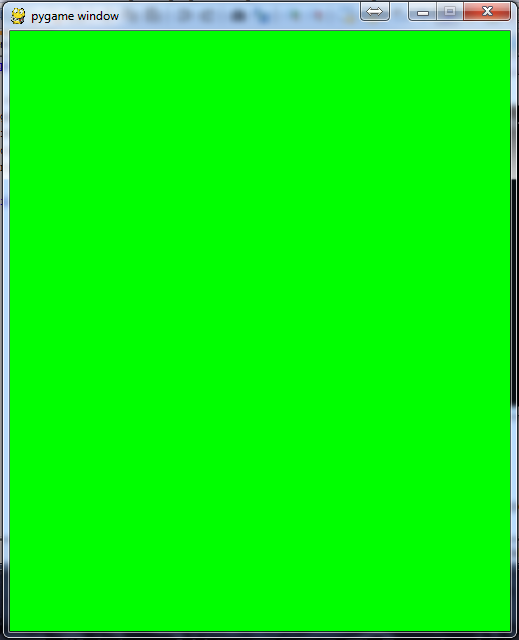
Starting off slowly with a general creation of the game screen the student is guided through a more procedural approach in video one with loops and if statements for event handling. With the second video the student is shown an easy object oriented approach which will tidy up the code from the previous video then, through some simple routines we can create the ball and have it moving and bouncing off sides of window. Next is the addition of the paddle and its user input for movement which can then be related to the ball’s position and if they hit each other the ball can bounce off. The bricks can then be added (and deleted if the ball comes into contact with them) – the ball must also bounce off the bricks. The engagement program will not be the full version of the game it has only basic interactions between the objects. The full version has different balls normal, fireball, fastball, extra balls; different paddles normal, shooting, small and large and different bricks normal special extra hit bricks. This program is to provide the foundations for the full version it should look and behave like the full version but must be uncluttered and easy to understand. An instructional video was created to guide students through the steps of creating the simplified version of the full game. The code is broken down into smaller segments adding elements and explaining what the code does and why it is used. I’ve tried to keep the videos as short as possible and tried to explain in as simple terms as I possibly could. This is after all a project to engage the students in programming not teach them the whole programming language.

**Video 1** [**- Creating a Pygame window and events for quit and pause**](https://www.youtube.com/watch?v=gXic9cbiXhY)**.**

Available at: <https://www.youtube.com/watch?v=gXic9cbiXhY>

The first video introduces the student to the construction of the game window, its placement and the event handling loops for pausing and exiting the game. It will also explain the os import for window positioning and key events like key down and key press. Simplicity is best as we want to engage not scare the students.

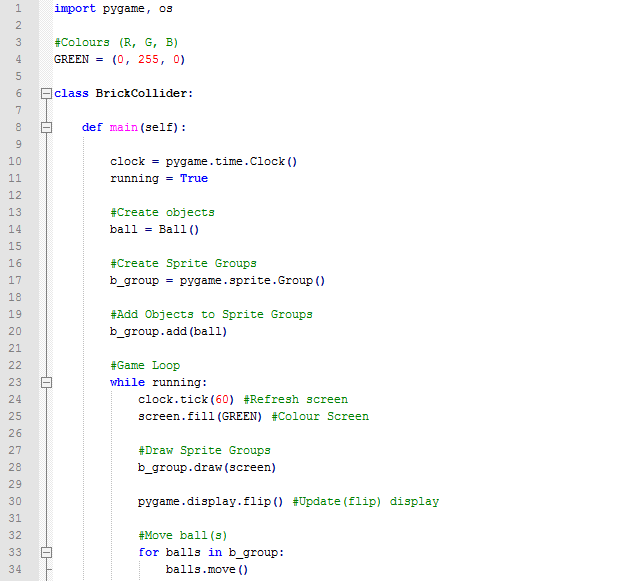
Code from completion of video 1.



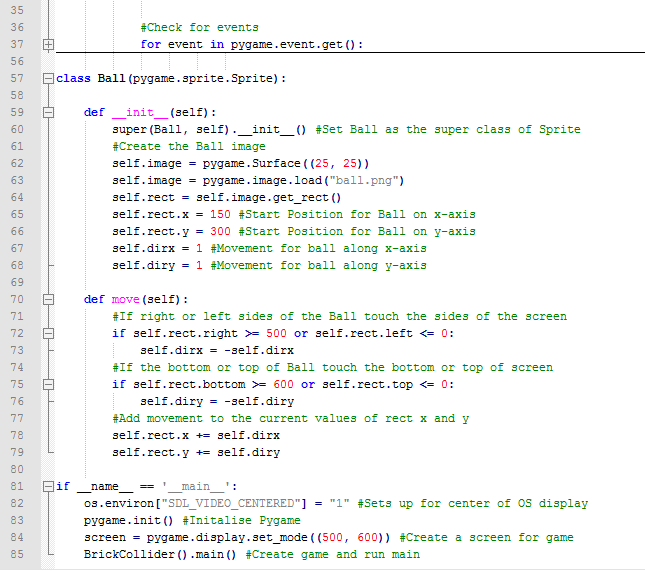
Game after video 1.

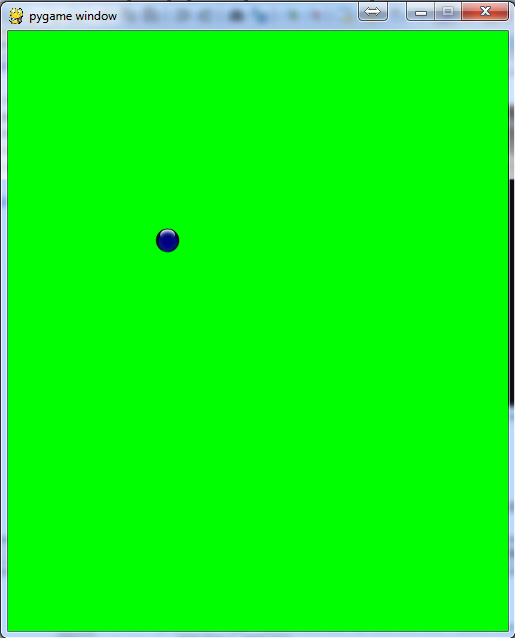
**Video** [**2 - Create the ball object and have it move**](https://www.youtube.com/watch?v=ARkWfDEFjCo)**.**

Available at: <https://youtu.be/ARkWfDEFjCo>

This video briefly introduces the student to Object Orientated Python. A change to the layout of code is needed to reflect a more Object Orientated approach. This segment shall include sprites and mentions inheritance and assigning the object to the sprite super class, without going into too much detail. The Ball Object inherits from the sprite super class and needs to be initialised as such. The Ball object creation with image, start positions for x and y axis and direction variables for movement on the x and y axis and functions to move the x and y position of the ball are all part of the Ball object and created in this video. Also the Ball needs to be drawn to the screen and added to a sprite group. The video finishes off with the creation of a colour constant value of GREEN which can now be added to screen fill instead of the R. G. B. values.

Code from completion of video 2

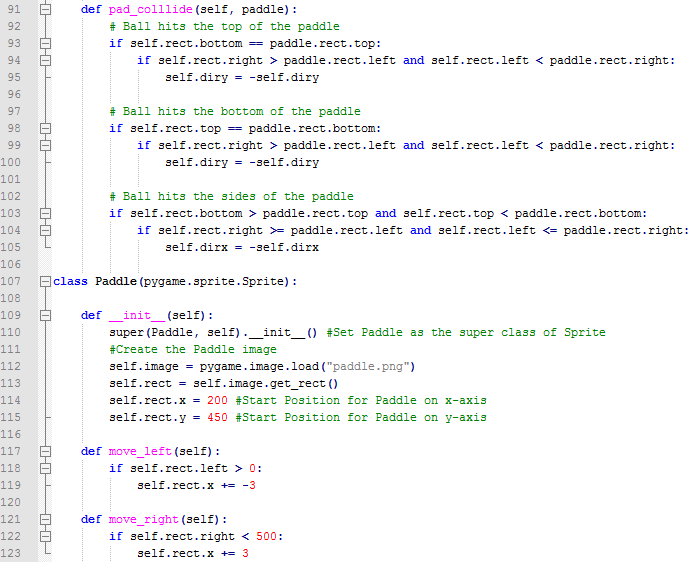
Code from completion of video 2.

****

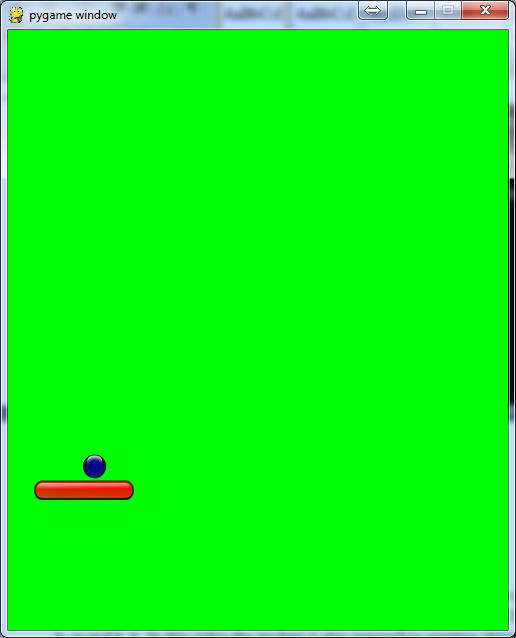
Game after Video 2.

**Video 3 -** [**Create the paddle, movement and interactions with ball**](https://www.youtube.com/watch?v=oYzJS-nBlXA)**.**

Available at: <https://youtu.be/oYzJS-nBlXA>

This video introduces the paddle to the game to send the ball back up the screen towards the bricks, which are created in the next video. Fairly similar to the Ball class it is also a sprite and will need to inherit from the sprite super class. The video guides the student through initial creation and helps create functions for movement of the paddle, left and right but not beyond the borders of the screen. A function is then defined in the Ball class to check if the ball collides with the paddle and if it does where and redirects the ball accordingly.

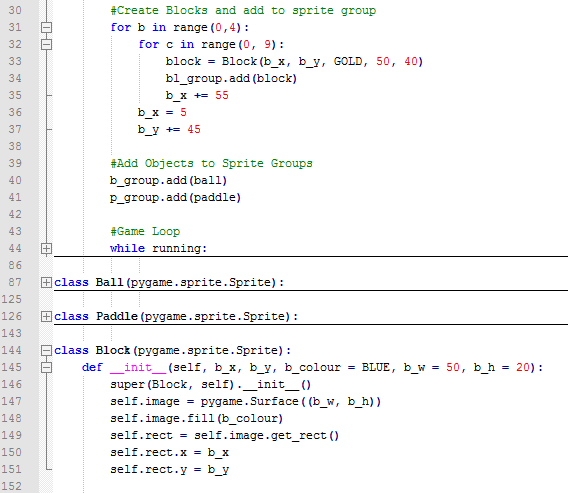
Paddle class with movement function and Ball function for collide.



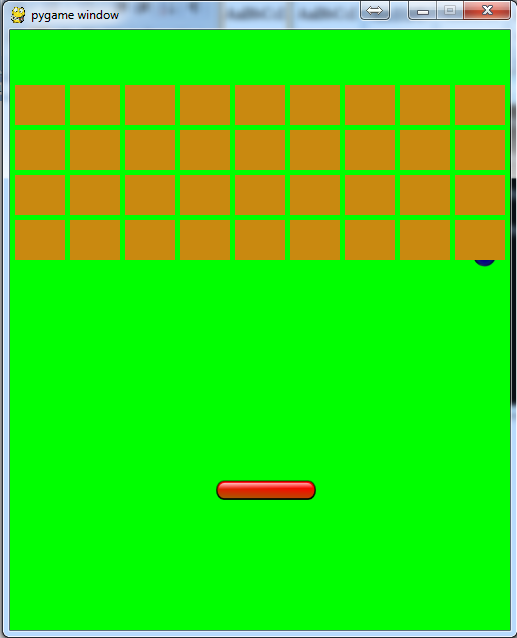
Game after video 3.

**Video 4 - Creating Bricks.**

Available at: <https://youtu.be/S3kJBCGU_Tw>

This video adds the Bricks to the game. Some defaults are used in the creation of Bricks in the Block class brick colour - b\_colour, brick width and height - b\_w and b\_h. In this video the student is also exposed to creating objects using default inputs and overwriting those defaults by adding arguments to the creation of the brick. A nested ***for*** loop is used to create 4 rows by 9 columns of bricks. To change the x and y values they are incremented within the loop.

Block class and for loop to create bricks.

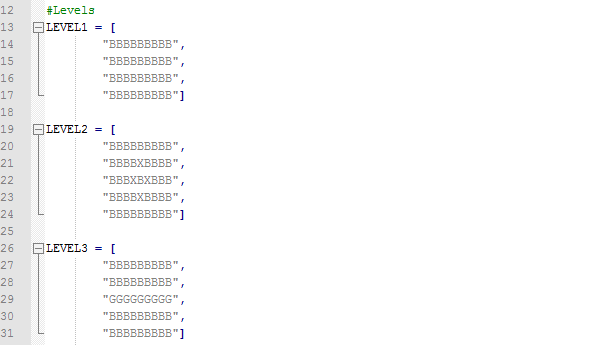


Game after video 4.

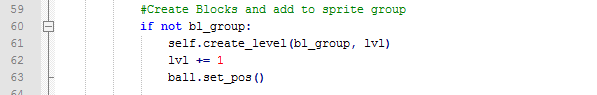
**Video 5 - Create Ball collide with bricks and 3 Levels.**

Available at: <https://youtu.be/vmr2VSPqt1M>

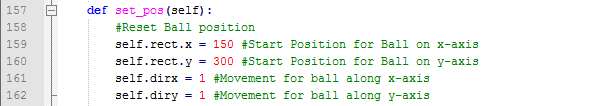
This video guides the student through the spritecollide function inherited from the sprite super class which checks if there’s a collision between the ball and any of the sprites in the bl\_group and removes the sprites (Bricks) from that group. It then shows the student how to create levels, each level is defined using a list and shown in human readable form, each row of the level is on a different line. The student is then shown how to determine the current level and create the bricks for that level, using a lvl variable and assigning the list for that lvl. The list is examined by iterating through each row with a nested ***for*** loop and a block created based on the letter in the list. If a block group is empty then the group is to be created. At the end of each level the ball needs to be reset for position and direction and is included in the video.



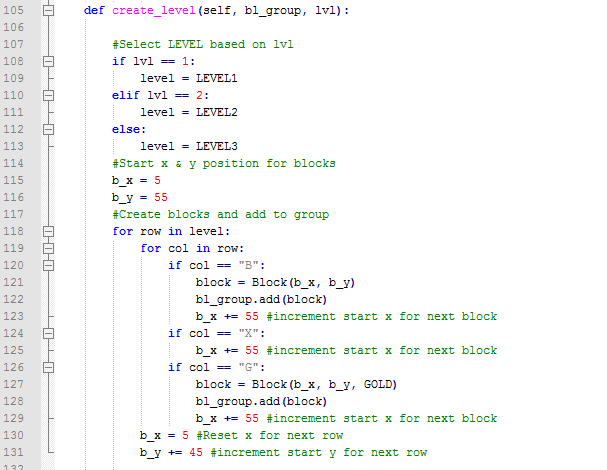
The LEVEL constants.



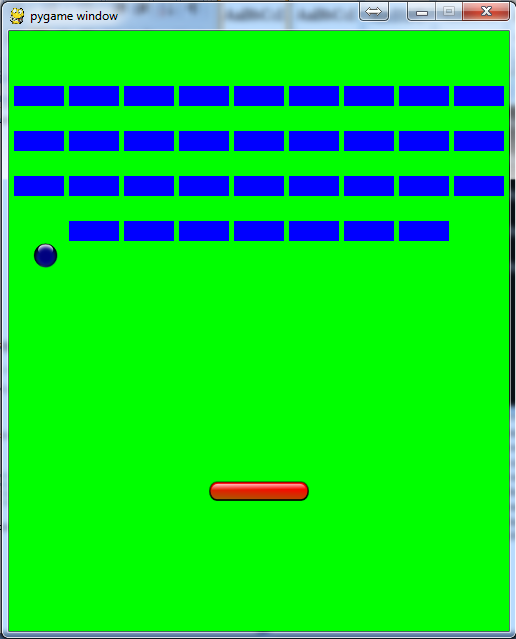
Call to create bricks and reset ball position.



The reset ball position function code.



The create\_level function code.

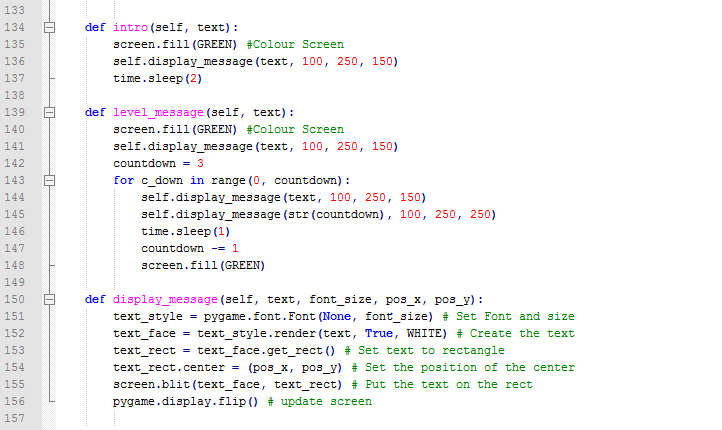


Game after video 5.

**Video 6 – Add text to game for intro screen and level countdown.**

Available at: <https://youtu.be/JKu5yQMzH6Y>

This video guides the student through adding text onto the screen to introduce the game and a countdown prior to the start of each level. Text is taken into a function and there font size and positioning is added, the default system font is used but the student can add a different system font if he/she should choose. The text is then put on the screen based on the centre of the rect. A countdown is added to create the countdown for the levels. The intro message is called for at the start of the main function so it won’t run at any other time other than the creation of the game and the level message is called for before the creation of the level.



Message creation functions.

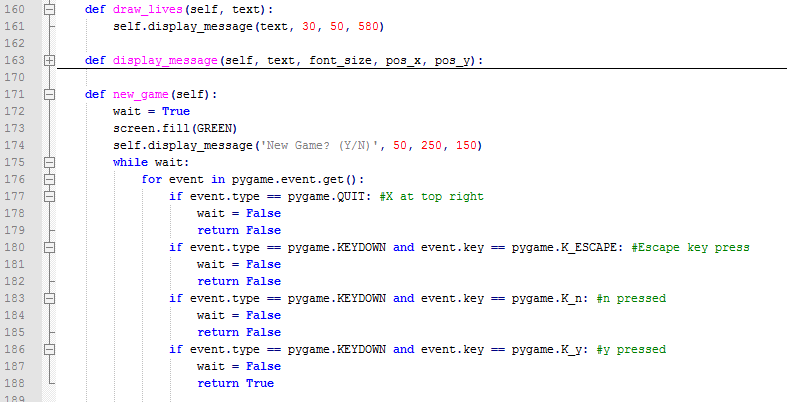


Game after video 6.

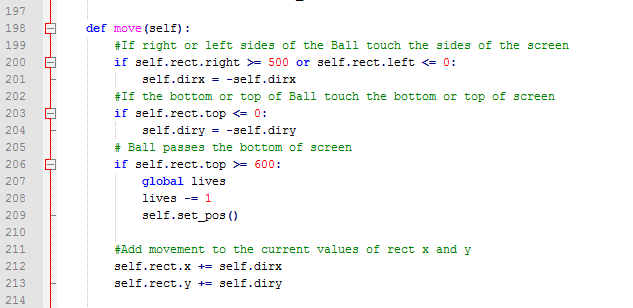
**Video 7 - Add text to game for lives count and a way to lose lives.**

Available at: <https://youtu.be/d8xzV-zZJ24>

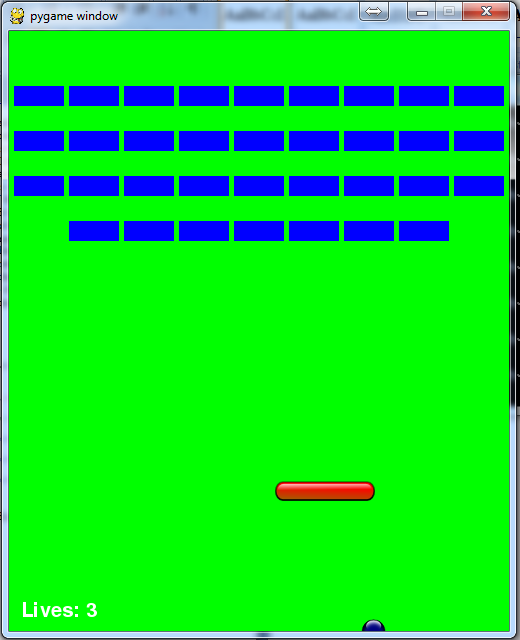
This, the last video in creating the basic version of Brick Collider, adds lives to the game, a way to lose them and a routine to restart game if the lives run out. A global variable is introduced in this video – the lives it needs to be accessed by the ball object and the BrickCollider object within the main function and game loop. This is the students first and only global variable in the game. The move function within the ball class will need altering to allow a condition should the ball pass the bottom of the screen, instead of bouncing off the bottom, the lives variable will need to be decremented by one and the ball position will need to be reset. Should the lives be less than or equal to zero a new function is added to give the gamer a choice to either start a new game or end the game entirely. An event handler is used in a ***while*** loop to wait for user input should he/she decide not to continue or use a way to close game the main game loop is changed to false and if they choose to start a new game the lives, level, the position of the ball need to be reset and the Brick group emptied. A simple function to display lives on the bottom left of the screen is also added.



Display lives and new game functions.



Updated move function to lose a life



Game after video 7.

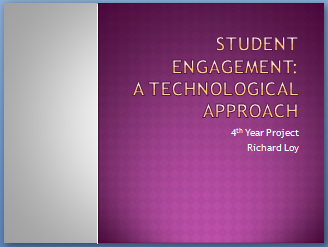
**Completing the game**

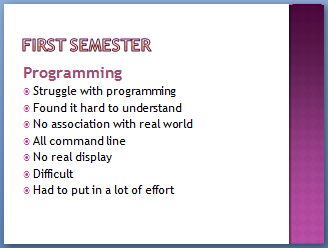
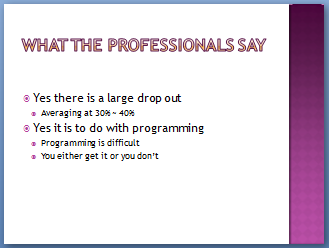
The game at this stage is functional and has only limited playability. It has only been used to engage the students in programming. They are free to explore further and add functionality themselves should they choose but for the purpose of completing the game I must add functionality to improve playability. Some ideas are:

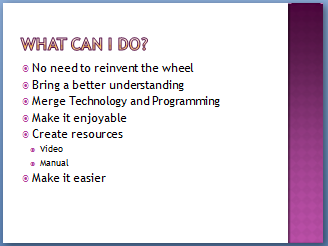
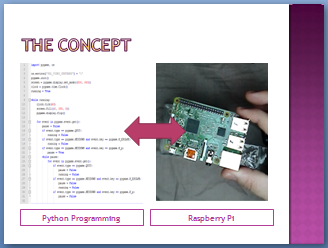
1. Bricks
   1. Harder bricks.
   2. Bricks that, are indestructible.
   3. Needs extra hits.
2. Special bricks containing mystery bonus/punishment.
   1. Extra ball.
   2. Fireball/Exploding ball.
   3. Fastball.
   4. Slow paddle.
   5. Longer paddle.
   6. Shooting paddle.
3. Paddle collision extras – different rates of direction.
4. More levels.
   1. **Appendix 5: Student Presentation**

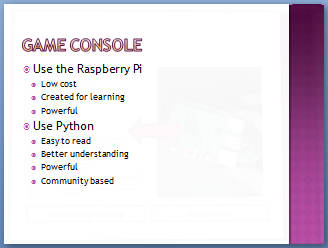
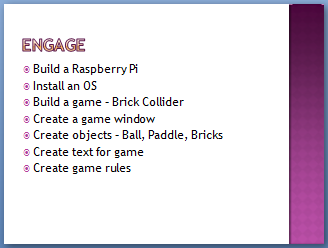
To gauge how the project would be received by students in their first year, I presented the project concepts to the current students at CCT. They have just completed their first semester and as such they would have just gone through the experience of learning programming for the first time. Throughout the presentation of the project the class were asked questions about the project and their thoughts and got to handle the Raspberry Pi to have a more thorough understanding of what the technology was.

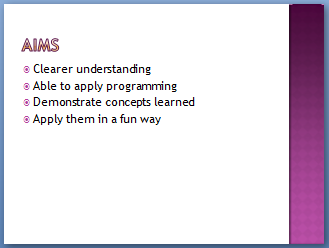
**Slide Show**

** **

** **

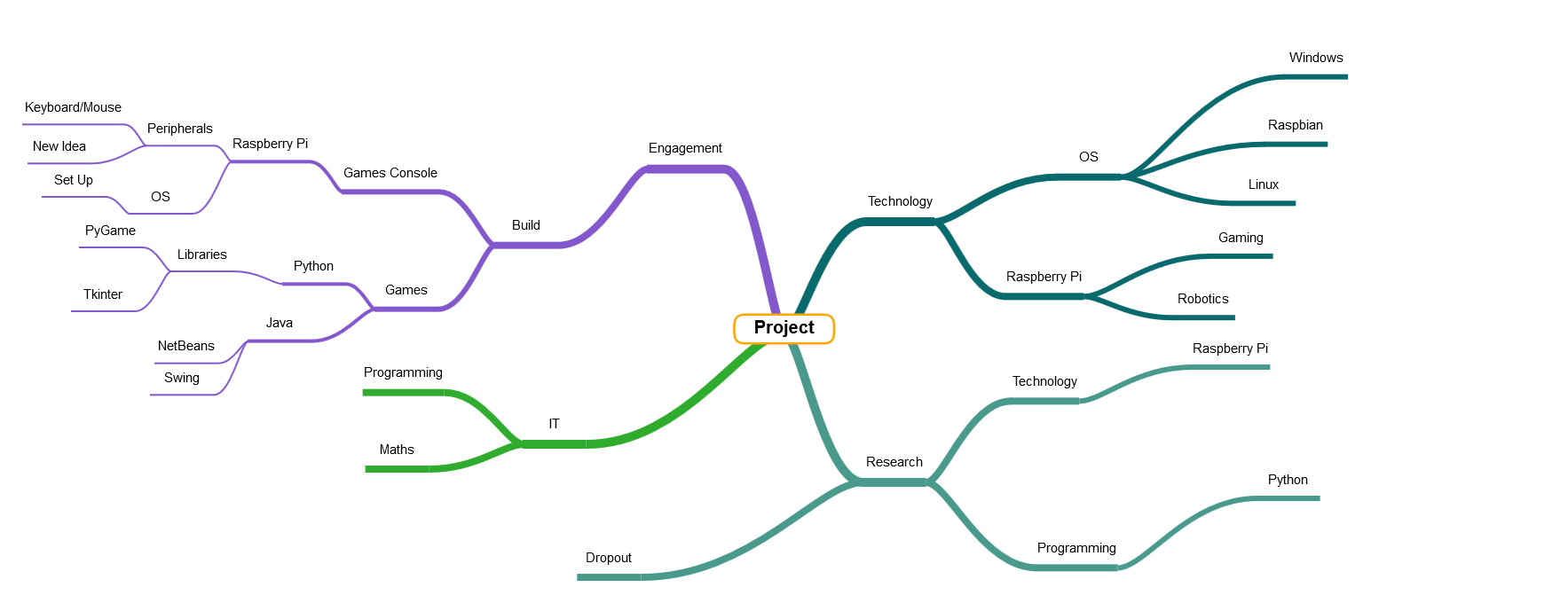
** **

** **

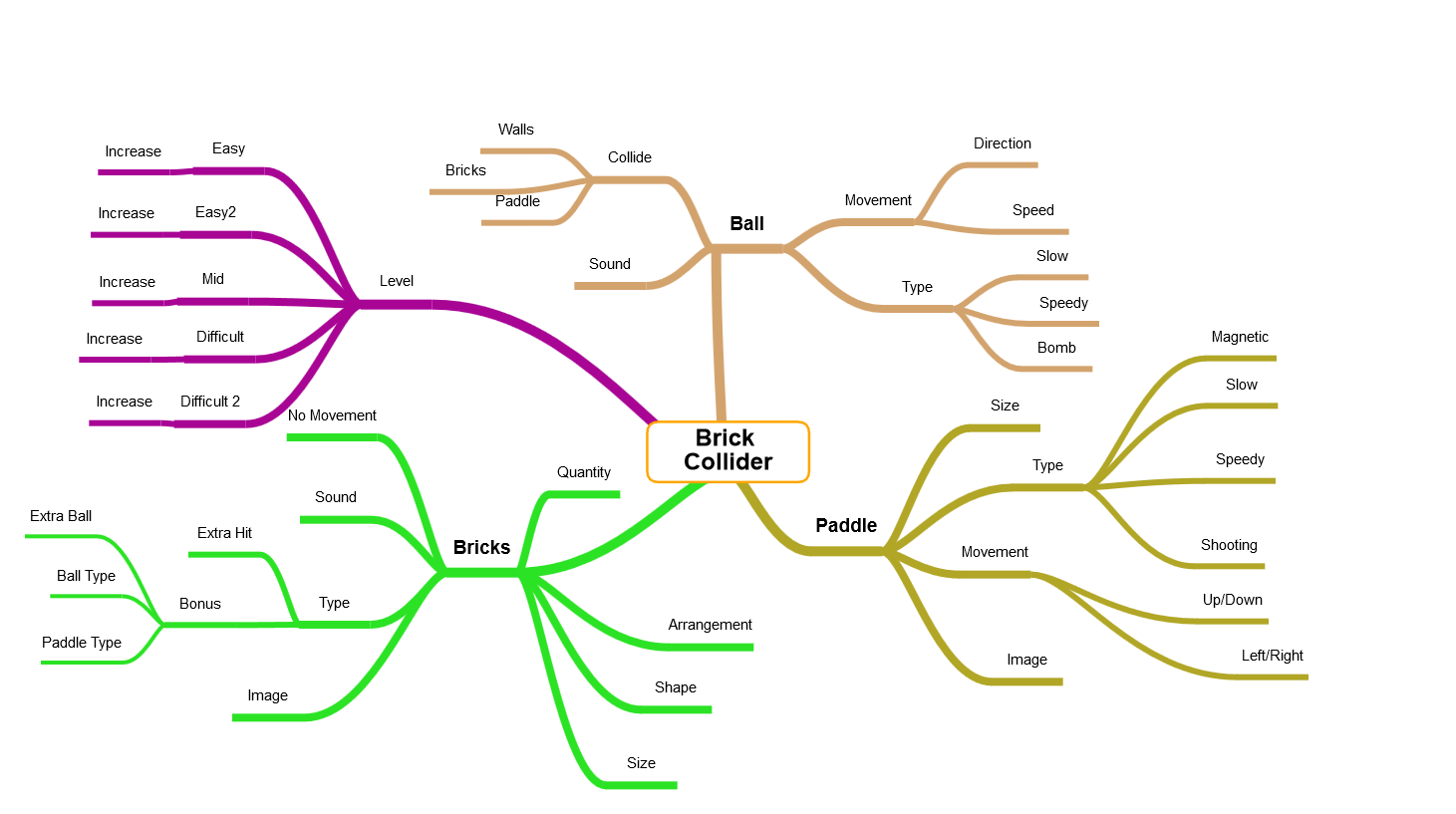
** **

* 1. **Appendix 5: Mind maps**

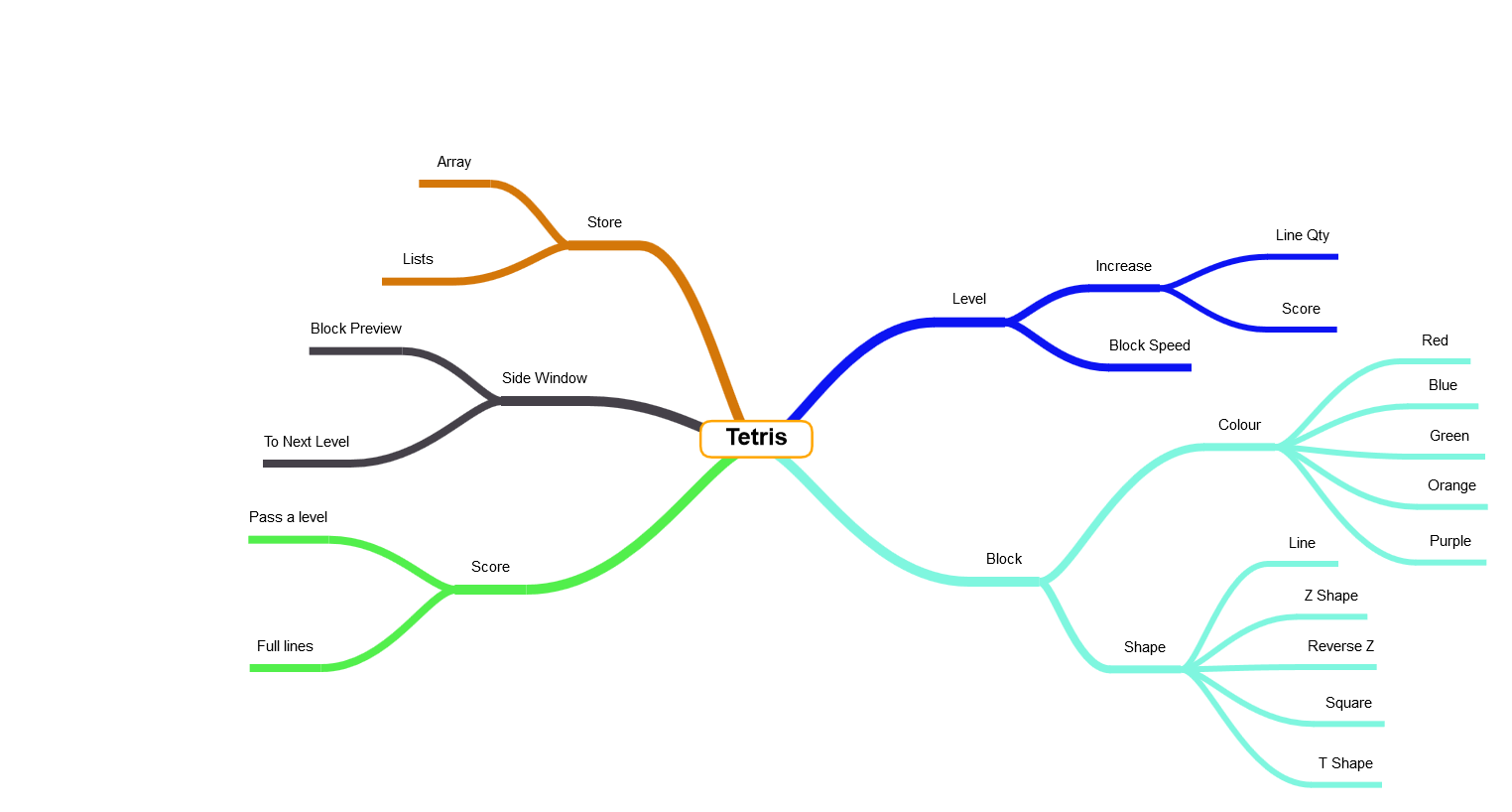
**Project**

****

**Brick Collider**

****

**Tetris**

****

**Mini Golf**

