<https://www.raspberrypi.org/help/noobs-setup/2/>

<https://www.raspberrypi.org/downloads/noobs/>

<https://www.raspberrypi.org/downloads/raspbian/>

<https://www.raspberrypi.org/forums/viewtopic.php?t=173955>

<https://gpiozero.readthedocs.io/en/v1.1.0/recipes.html>

<https://projects.raspberrypi.org/en/projects/physical-computing/8>

Arduino Project Elements

Description of original project concept:including features and limitations

(*what was it supposed to do what was it not supposed to do) 10%*

Timeline planning and final list of dates and what was supposed to happen when, and what did happen when. 5%

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Software Portion: The Python Game:** | | | | |
| **Task** | **Planned Start Date** | **Planned End Date** | **Real Start Date** | **Real End Date** |
| Proof of Concept | May 9 | May 11 | May 9 | May 11 |
| Initialize pygame | May 14 | May 14 | May 14 | May 14 |
| Create Sprites | May 15 | May 16 | May 15 | May 16 |
| Figure out how to render sprites | May 17 | May 19 | May 17 | May 20 |
| Start working on game loop and move player | May 20 | May 23 | May 21 | May 24 |
| Collision detection and crash function | May 24 | May 26 | May 25 | May 28 |
| Score Display Functions | May 27 | May 29 | May 29 | May 30 |
| Pause Functions | May 30 | May 31 | May 31 | June 1 |
| Game intro function | June 1 | June 3 | June 2 | June 3 |
| Convert program to support sprite lists | June 4 | June 6 | June 4 | June 8 |
| Make rockets work | June 7 | June 9 | June 9 | June 10 |
| Add Stages | June 10 | June 12 | June 11 | June 12 |
| Add Lives function (so player has 3 lives) | June 13 | June 15 | June 13 | June 14 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **The Hardware Portion: Raspberry Pi:** | | | | |
| **Task** | **Planned Start Date** | **Planned End Date** | **Real Start Date** | **Real End Date** |
| Proof of Concept | May 9 | May 11 | May 9 | May 11 |
| Install SD Card Formatter and format it | May 14 | May 14 | May 14 | May 14 |
| Install Raspbian on SD Card | May 15 | May 15 | May 15 | May 15 |
| Make a button and led circuit work | May 16 | May 22 | May 16 | June 10 |
| Hook up the circuit | May 16 | May 17 | May 16 | May 17 |
| Write Program | May 18 | May 19 | May 18 | May 19 |
| Troubleshoot | May 20 | May 22 | May 20 | June 10 |
| Add the rest of the buttons | May 23 | May 25 | June 11 | June 12 |
| Put the two programs together | June 15 | June 18 | June 13 | June 15 |

The project:

Photos: stages of your project. 5%

Wiring and schematic diagrams. 20%

Program code printout with comments and highlighting. 20%

Variable initialization

Setup and display

Collision detection

Class

Call function

Button processing

Learning paragraph for **each person in the group,** Paragraph should describe the part of the project you were responsible for. What skills or outcomes from the curriculum did you learn or practice and improve through your involvement in this project. 20%

Throughout this project I gained and improved on numerous skills through the usage of the raspberry pi whether that was the electronic or programming aspect. Electronically, I connected and tested button and led circuits using the GPIO module. Moreover, I used a procedural programming language, python, to apply fundamental programming concepts to develop a complex program that controls an external device using inter-device communication.

I encountered some issues, most of which were easily fixed, though it was sometimes difficult to identify what the problem was. One of these issues were that I was unable to render rockets for a lot time, until I changed my approach and started using sprite lists (“pygame.sprite.Group()”). When I began to use sprites lists, my problem was still unsolved so I decided to start all over again, and then just add all my other functions into my new program later. Eventually, I got it to work. Despite the time working to try to get the rockets to work, it was still worthwhile, because I learned a lot about different ways for rendering sprites in pygame. Overall, after all the struggles that I encountered during this project, it has taught me that all you really have to do to be successful in programming or life in general is never give up, because eventually I got my game to work mostly the way I wanted it to work.

Instruction list and need to know advice:

A list or paragraph outlining the most important things for someone else doing this project in the future. 20%

* Make sure you format the SD card before you upload raspbian on it
* Don’t pull the SD card out without shutting the raspberry pi down or ejecting it
* If the raspberry pi is stopping at the start screen, the SD card could be corrupted, so try re-uploading raspbian after formatting the card again.
* Use a header on the raspberry pi pins the ensure a proper connection is made throughout the circuit.
* Avoid short circuiting the raspberry pi as much as possible or else it will keep shutting down (try using resistors)
* The raspberry pi zero is smaller than the average pi, therefore it is very difficult to run a very complex program on it (eg. a game). Thus, the average pi is more suitable for the purpose of a game, due to it having greater processing power, but despite that, the user will still experience large delays
* Use a try and except statement to ensure that the GPIO pins are cleaned up before the program is killed.

If I had more time to work on this project, I would definitely try to make it so that there was more of a story to the ending of my game, in the sense that some kind of image would show that the captain of the submarine (the player) is giving the research to the scientists at Antarctica. Also, some difficulty that I experienced was that the edge of the shark or whale would collide with the submarine, even though, they didn’t appear to collide. Thus, I decided to leave it as a limitation of my game, because I was running out of time. The game could use a bit of touching up. Therefore, if I had more time, I would adjust the collision detection in such a way that it would be less frustrating for the player. Also, when a player paused or loses a live in my game, they have to use the mouse and then quick have to transfer their hand over to the keyboard, because the sharks and whale start coming into the screen almost immediately, which can result in another lost life and more frustration. My solution was to blit “3,2,1” to the screen, and use “time.sleep()”, however it is a little messy. If I had more time, I would try to change my program so that the delay after the pause and lost live function would be cleaner.

Code: