Natalia Palej

A00279259

Software Design with Artifical Intelligence in Cloud Computing  
Semester 1.2  
Year 1

Mobile Apps & Connected Devices 1.2

Pong Game

Table of Contents

[Introduction 2](#_Toc94819026)

[Context/Rationale 2](#_Toc94819027)

[Specs 2](#_Toc94819028)

[Design 3](#_Toc94819029)

[Hardware 3](#_Toc94819030)

[Software 3](#_Toc94819031)

[Main Game Flow Chart 4](#_Toc94819032)

[Lives Flow Chart 5](#_Toc94819033)

[Score Flow Chart 6](#_Toc94819034)

[Build 7](#_Toc94819035)

[Hardware/Software 7](#_Toc94819036)

[Test 10](#_Toc94819037)

[Conclusion 11](#_Toc94819038)

# Introduction

My goal in this project is to create pong game using Python Language and then running it on Raspberry PI and Sense Hat board (or Sense Hat Emulator). User will be able to move the bat with up and down joystick buttons on an actual device or by using up and down arrow keys if played on emulator. The game will cosist of a certain number of lives, a player’s score as well as increasing difficulty level.

# Context/Rationale

The logic behind this project is to expand knowledge on how to use Raspberry Pi joystick buttons, gain an understaing of loops in Python programming language, as well as become proficient with Python Functions (since many have been introduced to create the game).

# Specs

The user will be able to start the game by using one of the external sensor data (pressing and releasing middle button on SenseHat or hitting Enter button on the keyboard if Trinket Emulator is being used). At the beginning, user is given the instruction to do so by displaying a “Press To Start” message on the Sense Hat screen. Once the button is hit, user will come across few game objects:

* **Bat** - made of three pixels in length, white in colour and is located on the left of Sense Hat LED board (or top if Trinket emulator is used). The bat can move up or down by pushing joystick towards desired direction.
* **Ball 1 -** a single yellow coloured LED pixel. As the game starts, the ball moves to the opposite direction of bat. Once it reaches the end of the sense board, it bounces back and changes its moving direction.
* **Ball 2 -** introduced as the player progresses further in the game. To make the game more challenging and interesting, once the player reaches score larger than 5, an extra pink ball shows up on the LED Sense Hat board.
* Both, Ball 1 and Ball 2, increase moving speed once the score is larger than 10. The player is notified of increased difficulty level when the score is equal to 5, and then followed by 10.
* **Lives -** put in place to avoid endless game. The player starts with a total of three lives and can build them up once score 15 is reached. If the bat is missed, one live is deducted.
* **Score -** gained once the ball bounces back from the bat. The score amount also indicates the difficulty level that was mentioned above.

# Design

## Hardware

Raspberry Pi is an extraordinary device, it is known as a single board computer which means exactly what it sounds like: it is a computer just like a desktop but built on a single printed circuit board. Raspberry Pi is very small - roughly about the size of a credit card. Found at the top of the edge of Raspberry Pi’s circuit board and looking like two long rows of metal pins, the GPIO (General - Purpose Input/Output) header is how you can connect hardware like LEDs and switches to Raspberry Pi for control. In my project I made use of GPIO header by connecting Sense Hat to my Raspberry Pi. This enabled me to display images on the Sense Hat screen and make use of its sensors.

Sense HAT is an add-on board to Raspberry Pi that can be connected using the GPIO header mentioned earlier. It is made of 8x8 RBG LED matrix, six different sensors as well as a mini joystick. Some of the sensors it includes, but is not limited to, are accelerometer and temperature.

To further develop and test my application outside of college hours, I have used Trinket. Trinket is a page where you can write and run code in any browser on any device. Trinket introduced Sense Hat Emulator which I took advantage of to test my code changes when I did not have access to Raspberry Pi device.

## Software

Python is a scripting high-level programming language. Python supports libraries and packages, which encourages program flexibility and code reuse. To be able to control the Sense Hat add-on board I used a Python library called sense\_hat. This API allowed me to control RGB LED matrix and read the inputs from the mini joystick.

### Main Game Flow Chart

Diagram

Description automatically generated

### Lives Flow Chart

Diagram

Description automatically generated

### Score Flow Chart

Diagram

Description automatically generated

# Build

## Hardware/Software

Text

Description automatically generated At the beginning of my Python program, I have imported all necessary libraries (sense\_hat, time) and created an instance of SenseHat object. Afterwards I created three tuples to represent different RBG colours needed for my images that I will display on the matrix. Following that, I have declared and initialized variables that are necessary to start the coding process.

Afterwards, I created multiple functions to achieve desired gameplay. At the very start, I created a bat object than can be controlled by the user. The bat position spans to three pixels which are added to its primary location on y-axis (one above and one below).

Text

Description automatically generated

Following that, I implemented two separate methods for my first and second ball. The only variance betweem them is their assigned colour. First instance is yellow, and the second is pink. Draw\_ball method deals with reaching four ends of matrix board as well as collision with the bat. When that happens, it bouces the ball to opposite direction. It is also responsible for its velocity in a given direction.

Text

Description automatically generated

Text, letter

Description automatically generatedAs my next step, I declared two functions for the bat, which allow the user to move it up and down on y-axis. The funcions are in control of bat being drawn properly and not going outside the 8x8 board.

Once the game’s core functions were implemented and tested, it needed more challenge to expand the game and keep the user entertained. If the user misses the bat and the ball lands on x-axis (x = 0), one life is deducted and the ball position is restarted to its original location. When user loses all three lives, the message “you lost” shows up on the LED board. From that point, the game can be restarted. As a bonus, I have added supplemental ‘if statement’ that lets the player gain additional life once the score is equal to 15. That way the user can get the lives not only substracted, but also incremented.

Text

Description automatically generated

My last enhacement to the game was introducing scoring system. Each time the ball would be bounced off the bat, the player gains a score. As the game progresses, second ball is introduced resulting in increased difficulty. To further challenge the player, the speed of the two balls gets faster. The user is being notified of each difficulty incracement.

Graphical user interface, text, application, email

Description automatically generated

Lastly, to start the game I implemented method to start the game and an infinite loop that waits for the user to press and release middle button (or enter). If the user presses/pushes the joystick (or keyboard arrows) in any other direction it will print message with instruction.

Text

Description automatically generated

# Test

Graphical user interface

Description automatically generated with low confidence Firstly, I have pressed arrow button to see if user will be prompt with correct message and to make sure the game will not start. Once that was tested, I hit “Enter” to start the game. The bat and first ball showed up with no issues.

Afterwards, I moved towards testing the scoring and lives system. The score worked with no complications. While testing the game further, I came across one obstacle; Message in the console log was being printed multiple times. This issue is happening because my previously implemented function for tracking user’s score is being looped in the draw\_ball function. As the condition (eg. if score == 5) is met, it prints the message until score gets incremented. Same applies to extra lives message. Unfortunately, after multiple attempts, I failed to overcome this issue at this time. If I was to design the game again, I would undertake slightly different approach that would eliminate this problem altogether.

Graphical user interface, text, application, table

Description automatically generatedGraphical user interface, text, application

Description automatically generated

# Conclusion

Developing this project was most certainly challening and enjoyable. It activated my creative side and let me come up with various types of difficulties and additional functions for my game. What I enjoyed the most was testing the code during the labs. Seeing the program work was extremly satysfying. Working with a physical Raspberry Pi and Sense Hat makes me understand the code and funcions better, as well as how it operates – what happens when we change certain values. Even thought I can test as much on emulator, the experience is not the same. I am hoping to use my gathered knowledge from this project in my future development.