PyGame Coding Report

Callum Paton

December 2019

Eg19901@bristol.ac.uk

Throughout my project, I used my knowledge of data structures and loops alongside with new concepts from pygame that I accessed using various online sources. In this way I could build up complexities to form a completed and functioning game of 'Snakes'.

Developing a Window

Initially, I created a very simple window and ran a very simple game loop to develop my understanding of game loops and how to quit them using the built in QUIT function.

From here, I had to develop my understanding of classes. For my game, I essentially have two components. The moving 'snake' and the 'apples' that they eat. Each of these can be defined as individual classes. To develop my understanding of classes I referred to, https://www.programiz.com/python-programming/class and watched the following tutorial by 123 Animations.

Initializing the Player Movement

I then was able to define all the initial characteristics of the snake and apple in an __init__ function. For the snake, the next task was defining the move and collision functions. The movement works by adding x and y coordinates into a list with index 0 as the x coordinates and 1 as the y cooridantes. This led to issues initially as when snake is moving in a direction, the x or y elements of the list increase by 6 continuously in each direction. This led to a continuously growing snake rather than a moving block. However, using the .pop() function solved this by removing the last element of the list every time a new + or -6 coordinate is added. The basics behind this were learnt from, https://stackoverflow.com/questions/33537959/continuous-movement-of-a-box-in-pygame.

Collisions

The next thing I had to define was the game ending moves within the snake class. This involved collisions with the walls or collisions with the body of the snake. I used if statements, when the [0] (x coordinates) and [1] (y coordinates) exceeds the dimensions of the screen a collision is returned. I struggled with implementing collisions with the snake body, however, using my knowledge of the structure of lists I was able to resolve the problem. The [:1] – meaning from the index 1 onwards, meant that when the snake collided with any section that's isn't the head, a collision is returned.

Generating Apples

The next element of the game which is defined within a class is the Apple. These have two characteristics. The position on the screen and the regeneration of new blocks. I had to import random module to define the random position of the apple with random x and y coordinates.

Whenever the snake collides with the apples, a new one is regenerated. Initially, I had issues as the position of the apple was defined as a single point. However, this was eventually solved by using the abs function. This defines an absolute area that is the single point in the apple plus or minus the dimensions of the 15 by 15 block. Thus instead of being a single point, the snake can now collide with the whole apple and a new one is created.

The Game Loop

The game has 3 elements, key down events which change the direction of the snake, an if loop for generating new blocks and if loop registering collisions and ending the game. To implement a scoring system, I used my knowledge of while loops. I defined score = 0 outside the game while loop. Within the loop, whenever the snaked, moved to the food, score += 1. I then created a text displaying the score. Initially I had issues as I needed to convert my score to a string using the str() function.

Two Player

Due to my understanding of classes, I was easily able to make a two player version of the game by assigning a new snake to the player class. I had to have a separate loop to implement the WASD control. I also had to adjust the score system so the player who crashes loses 5 points.

Intro Screen, Pause Menu and End Statement.

In order to initiate which game loop to enter, I had to implement an Intro Screen. This loop is essentially a screen with a keydown loop. It asks the player to press 1 or 2 to go into the corresponding loop.

The Pause menu works in a similar way, when the user presses P they are brought out of the loop. Initially I had issues as whenever game was 2 player, they would be returned back into the one player version of the game. This was corrected using return.

Backgrounds and Images

https://stackoverflow.com/questions/28005641/how-to-add-a-background-image-into-pygame

Using this link, I developed my understanding of adding an image to a background. There are two images within my game, the background and the apple. I had difficulties replacing the rectangle with the apple initially. This was solved by the blit function, where the coordinates of the apple was simply Food.position.

Overall

Throughout my project, I developed understanding of pygame and referred to the links within this report and in my code. Specifically, my understanding of classes was crucial in this project. I also used my existing knowledge of loops and data structures e.g. using tuples to define colours and lists to create the snakes. Next time I would further develop my understanding of pygame and try to implement a computer controlled character.