Cian Hodnett – 121368013

CS2513 - CA2

‘Space Invaders’ project

# Features of my game:

* Player spaceship that can move left and right.
* Sprites and backgrounds.
* Collision on sides of screen to prevent player from leaving window.
* Enemies, which move to side of screen and then advance toward player.
* One projectile shot at a time, removes enemy, adds points.
* Collisions on enemies with projectiles.
* Score display in top left corner.
* Background music.
* Game over screen, shows current score, removes all enemies.

# Explanation:

## Player:

My code was to work, mostly, but not entirely in classes using OOP. I started with the players Ship class, and assigning x, y co-ordinates and ship size as well as its speed (vel).

I started implementing **the player sprite**, giving me a visual representation to work with. As well as a size reference for the rest of my game (48 pixels in this case). This would be implemented in a object under the ship class that would draw the sprites. Allowing me to later call this in the game-loop so it would be redrawn every frame.

I decided **the screen** would be square (48x48) and divisible by 13 sprites to allow for enough room to move and a standard by which other sprites would be displayed. This will come in handy for hitboxes later.

I then added functionality to **move the sprite left and right** by reading in the left and right arrow key presses and assigning an x-axis increase or decrease depending on the arrow pressed inside the game loop. The ship did not need a change in its y-axis, so I was able to set it and forget it for now. The y-axis would later, though, determine if an enemy had reached the player causing a game over or not.

The movement was much too fast, so I added a **cap to the frames** per second at 30 using pygame.time.Clock() and including the clock.tick(30) in the game loop. This made movement much more reliable and playable.

**Collision with sides of screen** was easy as a simple if parameter determined whether we had reached the sides of the screen and changed our x co-ordinates accordingly.

## Enemies:

Yet again, a simple case of creating a class and assigning a starting x and y coordinate as well as a velocity. Another simple if statement determined **if they reached the side of the screen**, then increased their y coordinate by 48px and inverted their velocity so they would head the other direction.

## Projectiles:

These were a little more complicated and required me going over the lecture on projectiles for help. These need several things, a list I could append and remove bullets from, to be drawn on top of the background, to append to the list when space was pressed and removed from the list once we had reached the edge of the screen, to collide with enemies by checking if the rect was inside the enemies’ square box of co-ordinates or not. I looped over the list the bullets were stored in and called my shoot function on them (applied velocity). This was by far one of the longer arts of coding this project and took many tries to implement correctly. In the end, turned out to be quite simple but required precise placement of some lines of code.

## Score:

This was a simple case of storing the score as an int value of 0 to begin with and **adding 5 every time an enemy collided with a projectile**. I used the pygame font function to initialise and display it in the top left corner, and chose the font impact because I felt it matched the game the most.

## Background Music:

This again was a simple enough feature to implement after some research from pygame documentation (<https://www.pygame.org/docs/ref/music.html>) and importing mixer. I simply gave mixer.music.play the parameter of -1 so the music would loop. (all sprites and music used are free for personal use and have been linked to and accredited in my comments on my code.)

## Game Over:

This was pretty similar to displaying the score on the screen but is only displayed when the enemy reaches the player, causing the game over state. Initially I tried a timer to close the game after being on the screen for 5 seconds but the time.sleep(x) did not appear to be functioning correctly on my end.