Tests Conducted

By Max Morphy

In order to test if the game is correctly detecting that a button had been clicked, and has also correctly detected which button had been pressed. I commented out the function that was meant to run when the button was pressed and used a console.log() statement as a flag that would output the name of the function run. This allowed me to test that the button press had been correctly detected and also that the correct function was promptly executed. Once detection was deemed to be working correctly, I added the function back in and tested that it was correctly performing its job as well. By the end of this process it could be determined that the process of detecting button presses and then executing the corresponding code is performed.

To test if the algorithm that detected if a bullet was outside the screen and then subsequently remove the bullet from the list of bullets, I added a console.log() statement that would print out the length of this.bullets I then played tested the game whilst inspecting the page so I could monitor the console. Initially, the console output was zero as expected, I then fired a single bullet and observed the console output being 1. I watched until the bullet left the screen and the console output went back to zero. This is a positive test, indicating that the algorithm was successfully detecting that the bullet was off the screen and then correctly removing it from the list and the length this.bullets return to zero.

When initially constructing the algorithm that would determine if two rects are colliding, I conducted a desk check to ensure the logic and control structures could correctly use the variables and input to accurately determine if the two rects were colliding. By carefully considering the five cases for rect collision in two dimensions (rect1 completely above rect2, rect 1 crossing the upper edge of rec2, rect1 completely within rec2, rect1 crossing the lower edge of rec2 and rect 1 completely below rec2) and similarly for the other dimension, I was able to determine that the algorithm could correctly and accurately determine whether or not two rects were colliding.

To test if my testCollision() method could correctly determine the kind of collision that had taken place, I added console.log() statements that would output “top”, “bottom” “left”, “right”. I then play tested the game and purposefully induced the condition that should trigger each of these cases and observed the output in the console section of the inspect element to observe if the game was correctly detecting each type of collision. Following this, I added in the code that would update the position and velocity of the moving rect. I then play tested again, and observed if the game was correctly updating the position and velocity of the rect in response to the collision.

I had some trouble initially getting the data from the json file to load in and be stored as variables in the format this.key = value; that could then be used through the game class. To ensure that the data was making it to the JavaScript program, I made the method responsible for importing data output the contents of the json file to the console. I also added another console.log() outside the method responsible for importing the data. I observed that the console.log() statement within the function importing the data could print the data but the console.log() statement outside the import function output the data as undefined. From this test I learnt that I need to assign the variables as properties of the game within the function in which they are imported so the program knows what data is being referenced.

As the application is a game, most features were tested and bugs discovered simply by playing through the game from the perspective of the user. One example of play testing I conducted was the addition of a double jump feature. I had some trouble adding the logic of double jump but once I did, I play tested the game and came to realise the way the game handled if the user was trying to double jump or not was not intuitive and did not feel right, In the end I removed the double jump as It did not feel good to use from the perspective of the player