Introduction:

My project was to re-create the famous arcade game Whack- A – mole using the Arduino Uno board. The idea was to build a 2-player game where each player has 1 red LED, 1 green LED, 3 general LEDs and a push button. The intuition behind this was to make each LED light up randomly and the players would have to push the button to whack the mole.

Main:
My strategy was to first create a 1 player game and replicate the same code for player 2. As the game must be played simultaneously, hence keeping the variables and times the same.

I first created a function called “randomizer_pi()” for each player which uses the random numbers generator where we assign random positions to the LEDs. Then we apply the “Whacky” function to allow us to turn on and off the LEDs for a random amount of time.

The game would only last until a player is the first to reach 10 points. This is indicated by making all the LED lights flicker on and off. To indicate that the game is over. To keep track of who is in the lead, I added a server and used it as a pointer. Which points to the direction to the player who has the highest score. This was easily implemented by installing the server module and points to the player who has a higher score. This was successfully achieved as I created a straightforward function that contained if statements. For example, if player 1 score was greater than player 2, then this causes the servo to rotate 180 degrees. If the players were levelled on points, then the pointer would point in between them.

As mentioned, I enabled each player to have their own green and red LED. The red LED portrays itself as a failure LED. This is because in my project I implemented a while loop function, where the chosen LED will have a given time range from 200 -700 milliseconds for being lit on. This function is what fundamentally makes it a whack a mole game. As it causes the players to stay alert throughout the whole duration, due to an LED randomly turning on with a volatile time range. Therefore, there will be times when a player is too slow to react to push the button. As a result of this, the red LED will shine soon after the random LED turns off, suggesting that they ‘missed it’. Which will cause their score to not be incremented. Hence, if this was the case then this caused the while loop to not be executed as the player didn’t satisfy the condition. I represented this with a Boolean flag for each player and equated it to 0.

If a player was able to successful hit the button within the duration of the LED being lit, then we would add a point to their score. Where both players had an initial starting score of zero. Which was initialized at the top of my code.

A way to improve my game, is to create a function that causes the random chosen LEDs to essentially be more random, although the random function is being applied. This is because out of the 3 blue random LEDs I noticed in my project that the left most LED for both players is being more favoured than the furthest right LED. This may be perhaps because once we restart the game then the same random sequence of numbers occurs again. Which defeats the game as players could play this game to infinity and would notice a pattern.

Furthermore, one bug I found is that my red LEDs which represent a miss did not light up when I pushed the button outside the duration of the LED being lit. My issue is that this successfully worked with the interrupt function in a one player but not with two.

IMAGES: