**Bop-It!**

**LAB 06**

**SECTION 1**

**Jesus Horacio Soto Gonzalez**

**SUBMISSION DATE:**

**10/21/2022**

# Problem

For this lab we are creating a program that replicates the game Bop-It! Using the DualShock 4 controller buttons. The program will continuously demand the user to input a correct and specific button as an input in a certain amount of time until failure.

# Analysis

The program needs to provide a starting menu and the user should access the game by pressing a specific button. The game needs to run until failure of the user to press the correct button or if the user runs out of time to input the required button. During the game the program needs to ask the user to input a randomized but specific button as the time decreases with every correct input. At the end of the game the program outputs the number of successful input that the user completed.

This program required some understanding of the data provided by the DS4 controller and how to interpret that data in order to develop the game.

# Design

For the design process of the program first, I made sure to define the number 1 as True, and number 0 as False, this was the base for the creation of the code since every button pushed will become a True (number 1) input.

After that I made sure to include all the variables required for the program which included all the buttons used for the game, the time of the round, the number of rounds played, etc.

After initializing the time of the round to 2500 milliseconds, I started the body of the code with a while loop. This said while loop houses the main many and the directions of how to start the game. I decided to use the button O as the input to start the game. If the user decides to start the game the input of the button O will become true (1) and the game will start.

Once in the game, I programed the code to randomly select a button to ask the user to input. I did this by using rand() and made sure that the range of numbers randomly selected started at 1 and ended at 4, since I assign every button a particular number in that range.

When the program randomly selects a number it is translated into a button that the user has to press in the button, the program then scans the input entered by the user and validates that it was pressed in the required amount of time with the help of another while loop, inside the previous one. Then the program comperes each button with an if statement and verifies that the value of the button required is 1 or true against the other 3 buttons.

If the user presses the corresponding button the game continues by asking for another input until failure as mentioned before.

Finally, at the end the user will see as a result the amount of successful number of buttons pressed.

# Testing

During the testing process of this project one of the main challenges that I encountered was how to start the program, I needed to find the correct way and process to help me develop the game. I decided to test different methods such as if, else if, else, while loops, etc. In the end I found that the most appropriate solution for this problem was to use everything. Using the knowledge that I have gained from this class I applied different methods in conjunction to make the program easier to understand and using while loops inside of more while loops with different if statements was a decent approach in my opinion.

Being able to reduce the round time each time the user selected the correct input was also a problem that I faced during testing but after observing the results I was able to solve this problem with patience.

Scanning for all the different data was another issue that I resolved testing over and over again until the desired result was reached. This due to the input of each button being compared to one another for the program to understand that the user made the correct selection.

Overall, the testing process went as expected, with some issues that I was able to solve with patience.

# Comments

This type of labs are the most interesting in my opinion since it uses all the different knowledge and skills that I have acquired in this class and makes us create something fun that we are familiar with.

Questions:

1. How did you randomize the buttons that needed to be pressed? **As mentioned in design I used a rand() function with a modulo 4 plus 1 to make sure that the only randomized values available were from 1 to 4. Similarly, each randomized number was paired with a button of the DS4 controller, and as a result when a randomized number is selected by the program a loop that ask for a specific button according to that number is shown to the user.**

2. What game states, if any, did you keep track of? **The main stats required for this program was the round time, buttons being pressed, and the amount of rounds being played.**

3. What mechanism did you use to make sure extraneous button presses were not registered?

**I used of the provided flag -b, and scanned the pressed of a button as an input of 1 or true and then compared them with the others at the same time to make sure that only the required one was being pressed. I did this by using and if statement with different or (||) logic operators.**

# Screen Shots

<Number the screenshots and paste here. The point of numbering the screenshots is so that you can refer to them during your discussion in the various parts above. Alternatively, you can include the screenshots in-line with the text above as part of your discussion.>