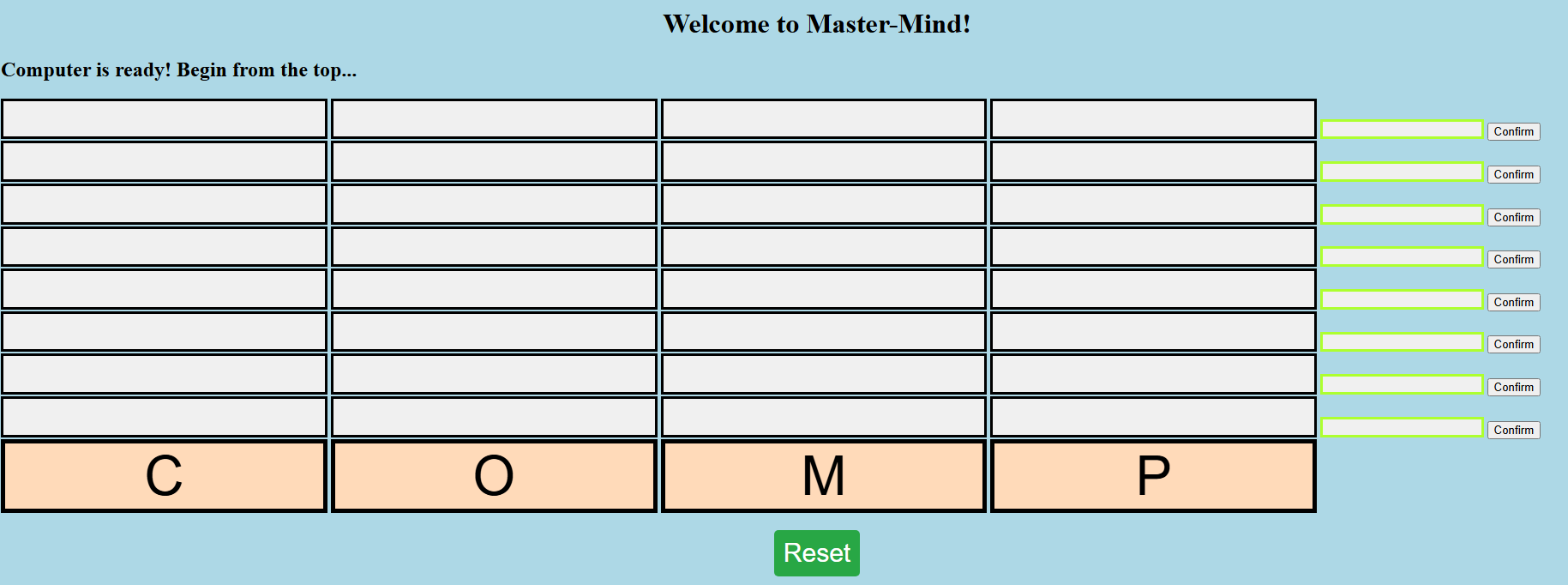
**Description**

****

Mastermind

For project 1, I have decided upon mastermind. The game was designed and implemented using a combination of Javascritp, HTML and CSS.

Mastermind is a code breaker game that was invented in 1970 by [Mordecai Meirowitz](https://en.wikipedia.org/wiki/Mordecai_Meirowitz), an [Israeli](https://en.wikipedia.org/wiki/Israel) postmaster and telecommunications expert. This is a childhood pastime enjoyed by my family. The rules of the game are as such:

1. Player 1 will select choose from 6 colors, and place then in 4 separate holes. Duplicates are allowed. Once selection is complete, he would raise a shield to hide his selection from his opponent
2. Player 2 will then begin guessing by also choosing from 6 colors to place into the a row of 4 holes in the board.
3. Once selection by Player 2 is complete, Player 1 would provide feedback by using black and white pegs. A black peg indicates a right color selection in the correct hole. A white peg would indicate a right color in an incorrect hole.
4. Based on the feedback, Player 2 will continue guessing until they either win, or they run out of rows for guessing.

The biggest difference against the traditional game is that we always play against human opponents, where it can sometimes be easy to look for a tell, or even hazard a guess based on the player's favorite color.

This project, would, however focus on a setting of player against computer. The most challenging portion turned out to be finding the logic to deduce which colors are guessed correctly AND in the right position, and also which colors are guessed correctly but placed in the wrong position.

**Link to the game:**

<https://weideow.github.io/Project1Mastermind/>

Instructions: Once the page is loaded, the computer player would have assigned 4 selections.

Player would begin from the topmost row.

Player can make their selection by clicking on the buttons of each row. Once selection is completed, pressing the confirm button would give feedback just like the black and white pegs in the traditional game. It will indicate how many selections have the right color and in the correct position, and also how many selections have the right color but in the incorrect position.

The game would end once player manage to guess all 4 selections correctly, OR when all 8 attempts fail the guess the right answer.

Player can choose to reset the game at anytime via the Reset button.

**WireFrame**

1. Upon initiation of game, play audio, computer to generate random colors in 4 different positions, which can also be duplicates. This selection must be hidden from player.
2. Set up a board layout, it should consist of 8 separate rows of 4 buttons. Each of the button would be able change to 1 of 4 colors. E.g. Clicking it once, would turn it green, clicking it again would make it blue, then red, then yellow.
3. Once selection of the row is completed, set up a button that finalize the selection (Confirm).
4. Once this button is pressed, there is no turning back! This would commence a loop through the 4 selections made by the player, which would then be matched against the computer's choices.
5. The computer would then flash the results. This would show, out of the 4 choices, which are BOTH correct in color and position, which are correct in color only, and which are totally wrong.

**Pseudocode**

1. Create an array containing the 4 colors for selection
2. For testing purpose, only 3 colors were used, and only 2 rows were created, with 1 additional row for the computer’s random selection
3. The rows shall be arrays containing empty strings, which would indicate the colors depending on how many times they are clicked.
4. Set up 4 clickable buttons for each row. Each click would bring up a different color using a loop.
5. Set up button that would end each round once selection is complete, to begin the loop.
6. The loop is designed to pick up all the colors selected by the player only, excluding the computer’s random selection.
7. Results would return after loop is completed to show the feedback.
8. If all the colors and positions match, end game. Otherwise, begin the next round!
9. Once round 8 is completed and game has not end, declare a loss.

**Planning Materials**

1. <https://pythontutor.com/>

This website enable me to visualize how the code works. Previously I was trying to piece together the solutions based off feedback from Stackoverflow, but was unable to understand how it works

**Attributions**

1. Picking random index, and mapping it to get randomly generated computer numbers

[https://stackoverflow.com/questions/69995256/loop-through-colors/69995422#69995422](https://stackoverflow.com/questions/69995256/loop-through-colors/69995422%2369995422%20)

This allows for looping over the colors endlessly with each click

1. Making checks to pick out the right colors, but in the wrong position

<https://stackoverflow.com/questions/27908135/mastermind-in-java-right-color-in-the-wrong-position>

1. Override Javascript in CSS

<https://stackoverflow.com/questions/5382254/how-can-i-override-style-info-from-a-css-class-in-the-body-of-a-page>

Learnt how to use !important to hide the computer’s random selections. Was previously trying to use compSelection.style.visibility =’hidden’. This command however always shows error in console.

1. Loop through the elements that are to show feedback (hints) to the player, and only populate them one at a time according to player clicking on confirm button

<https://stackoverflow.com/questions/60880521/how-do-i-loop-through-an-array-of-buttons-and-only-continue-looping-if-the-user>

This prevents the loop from populating all the feeback boxes, including the ones where player has not selected anything yet.

Next Steps