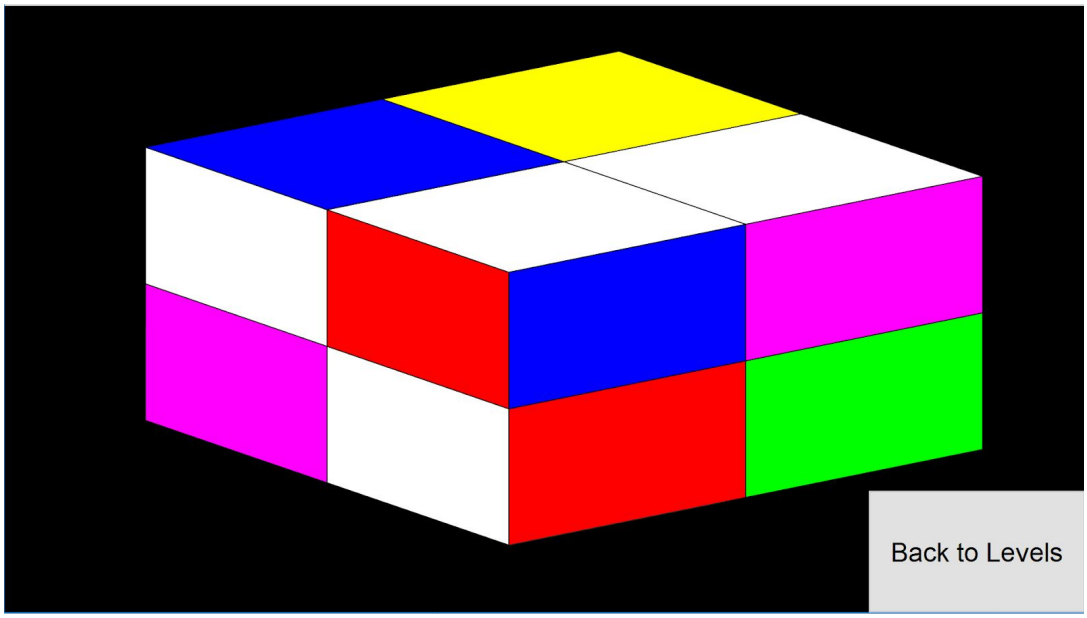
Beginner's Rubik's Cube
Final Project
Student: Melina Muthuswamy

Problem Statement: Create a program that dynamically allows the user to solve a 2x2x2 Rubik's cube through turning the faces of a properly visualized cube.



Outline of Working Principles:

The program will be divided into multiple tasks in order to most effectively achieve its purpose. The game has 8 levels which the user can choose to play, each with a different scramble of the 2x2x2 cube. 2x2x2 was chosen for this task due to the added complexities involved in solving a 3x3x3 cube and the time constraints of the assignment. Based on the level chosen, the program assigns scrambled colors to the 8 cube objects involved in visualizing the Rubik's cube. Object oriented programming was used in the solution to this problem in order to effectively and efficiently organize the data for each of the cubes. Each cube can be involved in three different directions of rotiations and therefore assigning a cube Object to each one with its unique properties made it easier to work with the cubes. The data of the different layers and the different degrees of freedom are arranged into various matrices. Based on where the user first clicks and then releases the keyPress, the program determines which direction the identified layer must be turned. The color assignments for each of the cube objects are then updated along with the visualization. Finally, when the user completely solves the cube, a final end game message displays and allows the user to pick a new level and continue the game.

Challenges:

Several challenges were encountered during the process of solving the Rubik's cube problem. One challenge was figuring out how to share the information of the first cube that was clicked with the information of the second cube that was clicked as they were both callbacks of keyPresses. In order to solve this problem, the callback functions were nested and the output of the first click was shared with the second click as an input to that callback function. This way all of the information was shared and could be used in subsequent functions. Another problem involved updating the cube object properties. After figuring out that all the cubes had to be stored in arrays, getting the 'colors' property of each object and in each array to update after each move proved to be more challenging than anticipated. In order to solve the problem, based on the type of rotation and the layer that was rotated, the global objects and the global matrices that they are stored in have to be updated after each move.

Omitted Features:

Due to time constraints, the functionality of giving the user a hint for the next possible move was omitted in favor of giving the user more options of scrambles to solve and practice solving the Rubik's cube.