**Walkthrough of the code:**

The way Mastermind works is that the goal is for the user to guess the secret combination. If the user guesses the correct color and correct position the code will output a black “peg” in the form of the letter “B”. If the user guesses the correct color but it is in the wrong position, the code will output a white “peg” in the form of the letter “W”. If the user guesses the wrong color and wrong position a blank “peg” is outputted in the form of the letter “X”. The user gets 10 tries to guess the correct combination or until they choose to quit and save the game. The user is prompted if they would like to save and quit after each turn. Each turn the user takes the guess and the console output for the feedback is saved to a text file. If the user wants to quit, a new file is created that saves the try the user quit on and the time(0) value that we used in srand(). This is important because if the user would like to load the game again, the time(0) value will let us create the same solution array and if the user opened the file with that number, they wouldn’t be able to understand the solution array. When the program is first run, the user is asked if they would like to start a new game or load a previous game. If the user would like to load a game, the LoadGame.txt file is loaded in automatically that has the try count and also the time(0) or seed value, as like to call it. After this is loaded in and the solution array is recreated, the code runs normally as if it were a new game. If you would like more in depth understanding of how our game works, please see the comments we added to our code.

**Classes:**

**Combination.h:**

This class was used to create the secret combination before the game starts. Combination.h utilizes the random function to select letters that correspond to the colors in the game. Each time the Combination class is used, a new solution is created. The random combination of integers is accessible from the class and used in the main code.

**LoadSol.h:**

LoadSol class is similar to the Combination class, but is used when the user loads a previous game file. This class is made to keep the original answer made by the Combination class from the user’s previous game. Without the LoadSol class, the secret combination will change into a different combination.

**MMclass.h:**

To reduce the amount of lines in the main function, we put the instructions for the game in a separate class file. MMclass.h explains the instructions of the game and is placed after the secret combination is set by Combination.h.

**Roles:**

*Apollo:*

Created the main mastermind file, file began with no classes and no File I/O.

*Bryan:*

Added classes to the main mastermind file and added code to the main mastermind file. Worked on final report.

*Zain:*

Added a feature to save and load a game using File I/O and working on the main mastermind file. Worked on final report.