Harkamal Grewal

CIS 37A Final Project/Poker Game Simulation

**Introduction**

Our Final class project is focused on building a card shuffle game that has the capabilities to shuffle, deal, and assess the better hand. Using the different starter coding stubs provided I had to determine the correct way to analyze a hand. It was necessary to print the specific set of cards that each hand was given. This allows for the program to judge which hand is better. The program itself is responsible for presenting two sets of hands and will determine the different components of each set. The program will output the different pairs the hand contains and will compare the results with the two sets of hands.

The pass by pointer technique played a major role in conserving memory consumption and being able to pass constant variable to access the random suits and faces. The shuffle method is specifically designed to select random suit and face numbers to store in the array of 52 cards which form a deck. After the 52 cards have been randomly set the dealing prints the first 5 cards based on the suits and faces given. After the deck has been set and the specific hand is put together the program will call different functions that will check the hand for specific pairs. Each pattern that is checked is then printed on the terminal for the matching pair. Two pairs of hands are generated from the same deck and compared. The comparing algorithm takes into account the number of pairs each hand contains. The hand with the most pairs is declared the winner of that game. Alongside checking the hand, the program will run 100 times and all the results will be stored in an array and copied into a file. The file will be generated at the same location of the program file and will display the winner of the 100 game simulation.

**Conclusion**

This project has taught me a lot about the fundamentals of C. I was able to learn how different methods and pointers interact with each other to complete a program. The point by reference technique is a very powerful in C and allows for the programmer to have complete control of the program. My approach to this problem was first understanding the given code and visualize its functionality and how the methods interact. Once I created the connection between the different features it was easy to add different functionalities. For future enhancements, I would really focus on the speed of the program as this program is inefficient and can be done at a better time and space complexity rate.