Joshua Ray CS 499 – T5477

Enhancement Two: Algorithms and Data Structures

The artifact used for this enhancement was a bot I made to run in Discord. The bot uses text commands and direct messages to play Texas Hold'em with other Discord users. I call this bot Poker Bot and I created it about 2 years ago as a side project. I selected this artifact because it was a project that I never finished but always wanted to. I also chose this because it was something I created before I had many of the skills I have now and would be able to easily find enhancements that would display my growth. Specifically, in the category of algorithms and data structures, the project lacks an algorithm for finding the winning hand of each round. I realized that the hands would likely have to be iterated over multiple times to find the best hand for each player. There would also have to be an algorithm for finding all possible 5-card hands out of the 7 cards each player has available. This seemed like the perfect opportunity to display my ability to use asynchronous code execution to evaluate the players' hand efficiently. Adding this enhancement would also lead to a finished, playable project.

I believe the enhancements I made in this artifact will display the course objectives I planned to meet. The use of algorithmic principles and computer science practices can be seen in the overall design of the scoring algorithm. Most of the code added can be seen in the Showdown case on line 467 of the PokerService, in the WinPot() method in the PokerService, and in the CalculateScore() method on the hand object. A combinations method is used to find every possible 5-card combination for each player. Each hand is then scored using an algorithm I created. These tasks are done asynchronously allowing for more efficient execution. I also added a CompareTo() method to my Hand object that allows the use of sorting that already exists in the framework. I realized that creating more asynchronous code led to less readable code. After determining that it would be unlikely for more than 8 players to be in a match at a time, I decided finding the combinations of the hands and calculating their scores could easily be included on the same asynchronous task opposed to separating them each to their own.

Making this enhancement gave me some hands-on experience using asynchronous programing. I had used the async and await modifiers in my projects in the past, but I did not know their meaning or how to use them efficiently. I have since learned their meaning and have read about how to use them correctly. This project allowed me to solidify my understanding of the async and await modifiers and when to use them. One of the biggest challenges to this enhancement was testing. I started by playing through quick rounds with friends to ensure the CompareTo() method was configured correctly. This was a tedious process and did not allow me to test the edge cases effectively since the deck was random each round. I decided it would be best to create a quick unit test project to ensure all scoring was created as intended. I created a single test that used multiple sets of arguments to ensure all tiers of scores are created as expected for each hand. This allowed me to quickly test my scoring algorithm and helped me find a few areas that needed improvement.