Project 1

Title

Poker

Course

CSC-17C

Section

47468

Due Date

November 30, 2025

Author

Ivy Fudge

Introduction

I am coding a simple poker game. I chose poker because I enjoy playing it a lot and I made a blackjack game last semester, so I was already a little familiar with the general flow of a card game. I spent about 20 hours working on it, with about 1000 lines of code, 3 classes – the deck of cards, the players, and the dealer – and an enumeration for the possible hand types.

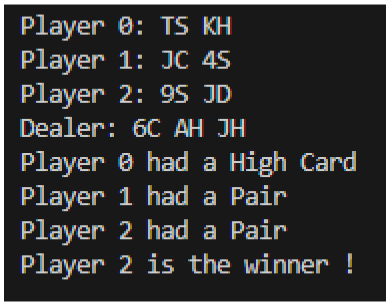
Game Rules

All players are initially dealt 2 cards each, then the dealer draws three cards for the table. Each player's score is then calculated based off of what hand type they are able to make using their cards and the table cards. The possible hands are Straight Flush, Four Of A Kind, Full House, Flush, Straight, Three Of A Kind, Two Pair, Pair, and High Card. To find the winner, the hands are first scanned for the best hand type. If only one person has the best hand type, they win. If there are multiple of the same type, then the hand scores, which are different per hand type, but are usually the highest face value of the hand, are compared. The person with the highest score wins. Now another game can be played.

Description of Code

The main file holds the game functionality and the game start. It creates a deck, 3 players, and a dealer. The player and dealer classes take a pointer to the deck, along with an ID number as arguments. The dealer class extends the player class to have a specialized show hand function. The deck is a queue of integers that act as the deck to pull cards from, with a set of the integers already in the deck. The player has a list of integers for its hand of cards, a counter for how many cards it has, a pointer to the deck, and an integer ID.

Sample I/O



```
Player 0: TS KH
Player 1: JC 4S
Player 2: 9S JD
Dealer: 6C AH JH
Player 0 had a High Card
Player 1 had a Pair
Player 2 had a Pair
Player 2 is the winner !
```

Checkoff Sheet

Container classes

Sequences

List: The player's hand is a list<int> to hold the card id's, which hold the card face and suit.

Associative Containers

Set: The deck uses a set<int> to hold the numbers already put into the deck when shuffling

Map: The player uses a map<HANDS, int> to organize the outputs of the function that calculates a player's hand type and score. This is then passed back to the main function to find the winner.