CS220 - Arch from a Prog. Perspective
Project 1

Due: 10/23/2022, 11:59pm

1 Introduction

In this project, you will implement a program that evaluates the winner of a 2-player Omaha-High Hold'em poker game (also known as pot-limit Omaha). The card game Omaha-High Hold'em poker is played using a standard 52 card deck where each player is dealt 4 cards and 5 shared "community" cards. Each player's hand is the best (i.e., highest ranking) 5-card combination that can be formed out of the 2 (out of 4) player cards and 5 community cards. Note that unlike Texas Hold'em poker, in Omaha you are required to use exactly 2 out of 4 player cards. Because this project deals with evaluating hands at "showdown" (i.e., at the end of a hand), for the purposes of this project, you are not required to know other details of the game such as how and when to bet, who bets, etc. A good starter video for Omaha can be found here:
`https://www.youtube.com/watch?v=gXoEtqm7Oos`.

In poker, hands are ranked, from lowest to highest, in the following way:

- **High Card:** Highest value card.
- **One Pair:** Two cards of the same value.
- **Two Pairs:** Two different pairs.
- **Three of a Kind:** Three cards of the same value.
- **Straight:** All cards are consecutive values.
- **Flush:** All cards of the same suit.
- **Full House:** Three of a kind and a pair.
- **Four of a Kind:** Four cards of the same value.
- **Straight Flush:** All cards are consecutive values of same suit.
- **Royal Flush:** Ten, Jack, Queen, King, Ace, in same suit.

The cards are valued in the order:
2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, King, Ace.

If two players have the same ranked hands then the rank made up of the highest value wins; for example, a pair of eights beats a pair of fives (see example 1 below). But if two

ranks tie, for example, both players have a pair of queens, then highest cards in each hand are compared (see example 4 below); if the highest cards tie then the next highest cards are compared, and so on.

A more elaborate description of poker hand rankings can be found here:
`http://www.pokerlistings.com/poker-hand-ranking`

2 Goal

The goal of this project is to write a program that given the hands of 2 players and the community cards, evaluates the winner. You will be provided with poker.c and poker.h files. You are to complete the specific tasks within the program, test it with the sample input and output.

Input: You will be provided with a plain text file that contains random hands dealt to two players. Each line of the file contains 13 cards (each card separated by a single space): the first four cards are Player 1's cards, next four are Player 2's cards and the last five are community cards. Each players best hand constitutes the best 5 cards from the player's hand and the community cards combined such that **exactly** 2 of player's cards are used. You can assume that all hands are valid (no invalid characters or repeated cards in the input).

Output: You are to generate Output.txt where each line contains the winner of the corresponding hand in the input. If the hand is a tie (see examples below) the output must indicate "No single winner".

3 Sample input and output

4 Provided Material

Other than this handout, you are provided with two files: `poker.c` and `poker.h` where each file has been marked with specific tasks you are supposed to implement. Additionally, you are provided with a reference implementation library (lib_poker_ref.a) that contains the reference implementation. In poker.c, you are to replace all calls to ref_*** functions with your own implementations. The reference implementation provides you a way to tackle each task independently. You are also provided with sample input and out output

Table 1: Sample input and expected output

Hand	Player 1	Player 2	Community	Expected Output	Reason
1	8H AS 3S 3H	TH AD 8D 9C	2C 9S JH 4D 5S	Player 1 wins	straight Five Four Three Two Ace beats two Nines Ace Jack Five
2	TC QC 9D 6S	5C 2S 6C 6H	6D TH JD 3S KH	Player 1 wins	straight King Queen Jack Ten Nine beats three of a kind Sixs with King Jack
3	7S 3H 5D 3S	KH 5C KS 9C	AH 5S 4C TH 7C	Player 2 wins	two Kings Ace Ten Seven beats two Sevens two Fives Ace
4	7D 5D 9H KS	7S 6H 3S 6D	TH 2S 7C 4D QD	Player 1 wins	two Sevens King Queen Ten beats two Sevens Queen Ten Six
5	7H KC JD 7C	6D AD TH KD	KH 6C QD 2D 4D	Player 2 wins	flush Ace King Queen Four Two beats two Kings Queen Jack Six
6	TC TD KH JS	8D JC 2H AD	4D 7S 8C JD 3C	Player 2 wins	two Jacks two Eights Seven beats two Jacks King Eight Seven
7	7S 9D 2C JS	AC 4S AD 2D	QC 3H KH KD 9H	Player 2 wins	two Aces two Kings Queen beats two Kings two Nines Jack
8	AH AC KH KC	AS AD KS KD	QC 3H TS 8H JD	No single winner	Both have straight Ace King Queen Jack Ten

files with 1,000,000 hands. Your code may be tested on additional inputs. The expected output must be **exactly** either `Player 1 wins`, or `Player 2 wins` or `No single winner`. However, for extra credit, the description does not need to be exactly as in Table 1 (that is, case of the text, prepositions, grammar, etc. don't need to be the same as in example). Yet, the description must fully describe the hand.

5 Submitting the result

- You must maintain your code on github with regular updates.
- You are free to work on the project on remote.cs or on your laptops/desktop, however, ensure that the project compiles on remote.cs

- **You are required to demonstrate your code to your lab TA. Appointment slots will be made available close to the submission deadline.**
- Other than the demo to your lab TA, you are required to submit your final code on Brightspace.

6 Extra Credit

There is a 2% extra credit for students who can also print out the player's best hand in text (see the last column in the table above). The project without extra credit is worth 15% of the course. With the extra credit, each member of the team can score a maximum of 17% on the project. Students willing to be evaluated for extra credit must email Professor Prakash for demo slots. During the demo, each member is expected to demonstrate thorough understanding of the design and code.