**CSC171 — Project 2**

**Poker**

# Purpose

The purpose of this project is to give you more experience designing object-oriented programs, and working with arrays.

# Problem Domain: Poker

Poker is card game which is thought to have originated in North America and has achieved global popularity. The game has a fascinating history which demonstrates it's profound impact on modern American culture. Some professors of computer science argue that the popularity of this national pastime is due to the interesting computational aspects related to the game. In this project we will use computer programs to examine aspects of the game. Similar to the first proejct, all that is required is a text based stystem.

# Problem 1: A Deck of Cards

Your first objective is to define a Card class which models the suit (hearts, diamonds, clubs, or spades) and rank (2, 3, 4, 5, 6, 7, 8, 9, 10, jack, queen, king, or ace). You will also need to define a Deck class which models a deck of all 52 cards using an array. You should be able to generate a new in-order deck, and you should be able to shuffle the deck into a random order. You should also be able to print an entire deck (and therefore any card).

To demonstrate your solution to problem one, write a short driver class which creates a new deck, prints it, shuffles it, then prints it again.

# Problem 2: Deal Some Hands

You should be able to generated two (5 card) hands for the two players by drawing from a deck. The key challenge here is to sample without replacement – the same card cannot appear twice. Write a short driver program to print the first hand, then the second, then the remaining 42 cards in the deck.

# Problem 3: Classify Some Hands

In the game of poker 5 card hands are ranked in order to determine who wins. Your program must be able to identify the quality or category of the hand. Traditionally, there are thought to be 10 categories, but there are really 9 (a "royal flush" is merely a special case of a "straight flush").

For this project, you should do some research into the game of poker to learn the rules and the hand rankings (you might even want to try playing a few games with your classmates). You need to identify the category of a hand - if a hand contains a only a high card, a pair, two pairs, three of a kind, a straight, a flush, a full house, four of a kind, or a straight flush. For conciseness, here are the classiciations in order:

1. Straight flush – all the same suit, all cards in order.
2. Four of a Kind
3. Full House – three of a kind and two of a kind.
4. Flush – all cards of the same suit.
5. Straight – all cards in order
6. Three of a Kind
7. Two Pair
8. One Pair
9. High Card

Note that if two players have the same level hand (e.g., both players have a pair) then the player with the higher hand wins. (E.g., a pair of aces beats a pair of twos.) If two players have the same pair (e.g., both have a pair of aces) then the game is decided by the highest card. To demonstrate your solution to this problem, you should augment your driver program from the previous question so that it also prints the classification of each player's hand. You should also print whether each player wins or loses the game, or if it is a draw. (In real poker you would have a round of betting, then allow the players to exchange some cards, and then another round of betting. For this project you are not asked to build a full poker-playing system.)

# Problem 4: Compare Some More Hands

Your program also must be able to identify which hand "wins" given two hands. You must write an additional driver program which reads pairs of 10-character hands from the terminal. Each string will be of the form RSRSRSRSRS where R can be any one of the set (2,3,4,5,6,7,8,9,T,t,J,j,Q,q,K,k,A,a) and S can be any one of the set (C,c,H,h,S,s,D,d). Your program must then classify both hands and print the winner.

## Documentation Requirements

In addition to submitting Java programs, you are required to submit one text document (named README) which specifies for each problem:

1.A one paragraph description of how you solved the problem 2.Detailed instructions on how to run the problem (What command arguments/input you give to it, what output the program prints out, what the outputs mean, etc.) 3.User created test cases, if any.

## Submission

The project submissions will be received via blackboard. Late submission will recieve a -2

You are expected to submit:

Java source files and compiled Java class files. You may also submit any additional files needed for running the programs. Documentation, described in the section above. If you submitted any additional files, explain what they are and where they are used.

## Grading

Evaluation for each problem will be divided into test cases, code review and documentation. The TA will compile and test your program with a few test cases, and also with any test cases you provide. If your program only produces partial results, providing your own test cases will give you partial credit for the parts that operate correctly. The code review will be based on evaluating the correctness of your code.

## Late Policy

As detailed in the course syllabus, late projects are assessed a 2% per hour penalty (so approximately 2 day maximum).

# Collaboration Policy

You will get the most out of this project if you do the work yourself. Watching someone else program is no substitute for doing it yourself. Doing the projects yourself will improve your performance in the rest of the course.

However, as detailed in the course syllabus, collaboration on projects is permitted, subject to the following requirements:

- Groups of no more than 3 students, all currently taking CSC171.
- You must be able to explain anything you or your group submit, IN PERSON AT ANY TIME, at the instructor's or TA's discretion.
- One member of the group should submit on the group's behalf and the grade will be shared with other members of the group. Other group members should submit a short comment naming the other collaborators and which of them submitted the actual project.
- All members of a collaborative group will get the same grade on the project.

Any other submission of code that is not your own constitutes a violation of the University's Academic Honesty policy.