Notes:

* Please put your name and the date, as well as the name of any collaborators at the top of each problem in a comment. Comments are lines in your program that begin with “//”
* Absolutely NO plagiarism. Each program should be your own work and not include any imports we haven’t used in class. You may talk to your classmates or to me (encouraged), but the program you turn in should be 100% written by you. I reserve the right to ask you what your program is doing, and you should be able to justify what it is doing, how, and why.
* Each problem should be in a SEPARATE class within the same package.
* Due date: Thursday, November 1 by the start of your lab section. Please email your .java files to chris.cain@rocky.edu.
* Thoroughly comment your code. For these programs, have at least as much English in your comments as you have Java within the main function.
* If your program does not compile (it has errors and can’t be run as is) you will score at most half of the total points for that problem.

Problem 1 (100 points): Simulate a deck of cards by creating classes for a card, for a hand, and for a deck. A card should have a field for a suit and a value, a constructor that takes a suit and value, and a method toString() to output the card to the console (eg. “Four of Spades”).

A deck should have a field for an array of cards, a constructor that initializes each of the cards, and a method to shuffle the deck.

A hand should consist of an array of 5 cards. The constructor will take a deck of cards and draw a hand of 5 cards (look at the first 5 elements of the deck). For this class you’ll also write boolean methods for each poker hand, returning true if this hand satisfies the conditions to be that hand, and false if it does not.

From these tools, write a main method that gets input from the console, specifying the number of simulations to run. From that, create a deck of cards. Within a loop, shuffle the deck, draw a hand, and call a function that takes a hand as input and returns which poker hand it is (by calling the appropriate methods in the hand class). After the loop terminates, calculate the probability that each hand is that particular poker hand.

Example:

Please enter the number of simulations

10000

Royal Flush - 0%

Straight Flush - 0%

Four of A Kind - 0.02%

Full House - 0.15%

Flush - 0.2%

Straight - 0.39%

Three of A Kind - 2.08%

Two Pair - 4.76%

One Pair - 42.32%

Nothing - 50.08%