CSC 260.002 Programming Assignment 5

Due Date: Thursday, October 6

You will implement **two programs** in this assignment, both require the use of methods.

Program 1: password tester system

Use a data verification loop to input a password from the user and repeat until the password passes four tests:

1. It is long enough (8 or more characters)
2. It does not start with a digit
3. It is not too repetitive (there are no occurrences where the same character appears three or more times in a row)
4. It contains at least one upper case, one lower case, one digit and one punctuation character

Create four separate methods, each to test one of the above requirements. Each method is passed the String password and will return a boolean where true indicates that the password passed the given test. Your main method will create a Scanner and enter a data verification loop which will input the password and call each of the four methods until the password passes all four. If any of the above tests is violated, output a message to indicate the problem. You may do the output either in the method or back in main, your choice. After the loop, output a message that the password is accepted. Example output is shown below. In the 3rd example, the password violates two rules so both violations are output. Note that the second output (not enough diverse characters) is actually 1 line of output but shown on two lines here to fit in the width of the document.

```
Enter your new password:  aaa123AAA#!
Your password is too repetitive
Enter your new password:  1234AaBb*
Your password starts with a digit
Enter your new password:  a123
Your password is not long enough
Your password does not include enough diverse characters
   (uppercase, lower case, digits, punctuation marks)
Enter your new password:  a123AB!12
a123AB!12 has been approved
```

To test to see if a character is upper case, lower case or digit, use the Character methods from chapter 4 (Character.isUpperCase(ch), Character.isLowerCase(ch) and Character.isDigit(ch)). If it is none of these, assume it is a punctuation mark.

Run your program multiple times using different passwords to demonstrate that it works correctly by inputting passwords that violate one or more rules and passwords that are correct. Make sure to test every one of the four rules. Collect a representative subset of input/output to attach to your program (or include in comments). Make sure your output shows some passwords that violate multiple rules and that you have at least one of each kind of rule violation among your input/output.

Program 2: Poker Hand Evaluator

Write a program to generate a “3-card poker hand” by randomly generating 3 int values from 0 to 51, convert each int value into a String description of the card (described below) and then output the best poker hand that can be formed out of the 3 cards. NOTE: using a random number generator, we have no guarantee that the same number is not generated twice out of three, but we will not try to prevent this – we will learn how to avoid this when we cover arrays. The program will be written in methods as follows.

1. A method to randomly generate and return an int number from 0 to 51 (pass the Random object to this method)
2. A method to convert an int number that represents a card into a String descriptor of the card (e.g. “5 of Spades”), this is explained below (pass this method an int and it returns a String)
3. A method to determine if the three int values representing a hand are any of 3 of a kind, 2 of a kind, or 1 of a kind (pass in the 3 int values and return 3 for 3 of a kind, 2 for 2 of a kind or 1 for 1 of a kind)
4. A method which is only called if the hand is 1 of a kind to determine the highest card value and return it so that you can print out “King high” rather than “1 of a kind (pass in the 3 int values and return the largest value as a String such as “King” or “9”)
5. A method to determine if the three cards are a straight (in numeric sequence) (pass this method the three int values and it returns a boolean)*
6. A method to determine if the three cards are a flush (same suit) (pass this method the three int values and it returns a boolean)

The program should output the 3 cards and based on the results of methods 2-6 output the hand’s value (flush, straight, 3 of a kind, 2 of a kind or high card). Only output the highest value of the hand including if it has is a straight flush (only output flush). * - method 5 is challenging, see below. If you successfully implement it, you will receive extra credit. If you cannot successfully implement it, have this method always return false to complete the assignment.

To convert an int `value` into the String description of the card use the following logic:

```
faceValue = value / 4 + 2
```

If faceValue is between 2 and 10, then convert the int value to a String as follows:

```
faceDescription = “″ + faceValue;
```

otherwise the faceValue is 11, 12, 13 or 14, convert these to “Jack”, “Queen”, “King” and “Ace” respectively

```
suitValue = value % 4
```

If suitValue is 0, this is “Clubs”, 1 is “Diamonds”, 2 is “Hearts” and 3 is “Spades”

Determining if 3 cards is a straight is complicated. I suggest the following approach. First, take the three int values and divide each by 4 and add 2 so that you have the three cards’ face values (you don’t actually have to add 2 for this if you don’t want to). Next, store the three values in three local int variables which represent the smallest, middle and largest of the three values. There are 6 combinations of this, you can use 6 if statements or some nested if-else logic to handle this. The 6 combinations are shown below assuming the 3 values are stored in x, y, z.

x, y, z   x, z, y   y, x, z   y, z, x   z, x, y   z, y, x

Let’s assume you have ordered these in the variables small, medium, large. You have a straight only if small == medium – 1 and medium == large – 1. For simplicity, we will assume Aces are high cards only.

Once your program is running, place a for loop in your code around the methods to generate a hand, determine its value and output the hand and its value so that you can test it out and obtain a representative set of output. You might use a for loop that iterates 10 to 20 times. What follows are some example outputs of running my program.

```
Your hand is 8 of Diamonds, King of Hearts, King of Clubs
2 of a kind

Your hand is 9 of Clubs, Queen of Clubs, 7 of Spades
Queen high

Your hand is King of Hearts, King of Spades, King of Spades
3 of a kind

Your hand is 6 of Hearts, 4 of Hearts, 9 of Hearts
Flush
```

Collect a random sampling of output demonstrating at a minimum four different types of hands as shown above. Note that straights are the hardest to come by so you may not generate a straight after several dozen attempts. Collect between 4 and 8 sample outputs and attach them to your program (or put them in comments).

Hand in both programs with their outputs.