Nassau Community College **Summer I 2016**
CSC 120 CA: Computer Science I **Prof. Christopher R. Merlo**

Lab Assignment #7: Writing a Data Type Class
Due Thursday, 9 June 2016 10:05 am

Please help NCC conserve resources by not printing this document on campus.

Objectives

- Write a data type class that uses another data type class
- Test a data type class by writing an application class

Partners

You may choose your own partner for this assignment, and I strongly recommend you do. Before you start to work on this assignment, take a look at how much there is that needs to get done. At least look at the 136 lines of output in the expected output file (download it from the Lab Assignment's web page). If two students share the workload, each writing about half of the required code, they will probably get done before leaving tonight. I am not honestly sure, however, if one student can get all this done by him/herself by tomorrow morning.

Overview

You've been using a pre-defined `PokerHand` class for a while; now it's time to write its code yourself.

Assignment

Several of these instructions will require reading the comments in the `PokerHand` class before attempting to write the required code. I encourage you to ask questions along the way, but please be aware that if you ask a question that makes it obvious you haven't read the comments, you may receive an unsatisfactory answer.

Setup

1. Create a new project named `Lab 7`, a new package inside the project called `lab7` (using all lowercase letters), and create three classes in the new package: `Card`, `PokerHand`, and `Lab7App`.

2. Download the files `Card.java`, `PokerHand.java`, and `Lab7App.java` from the lab assignment web page, and move them from your Downloads directory into the directory that contains the files you just created, overwriting those files. The `Card` class is complete. You will be writing code for the `PokerHand` class, and adding code to `Lab7App` to test `PokerHand`.

3. Modify the comment at the top of `PokerHand` to include your name.

4. Write the parameterized constructor in `PokerHand` to conform with the comment provided.

5. Modify the comment at the top of `Lab7App` to include your name.

Four of a Kind

6. Notice that the `Lab7App` class consists of a `main` method that just calls other methods in the same class. Each of these other methods tests one facet of the `PokerHand` class. Each method has already instantiated all the `Card` objects you'll need to test the methods.

   In the `testFourOfAKind` method, create a `Hand` object that contains all the twos – the two of clubs, the two of diamonds, etc. Call the `Hand`'s `toString` method; you should generate the following output:

```
Testing fourOfAKind method:
2 of clubs
2 of diamonds
2 of hearts
2 of spades
```

7. Now call the `fourOfAKind` method on the `Hand` and, using an `if` statement, display either "`Four of a kind`" or "`Not four of a kind`".

8. Use the `PokerHand` class' `replaceCard` method to replace the second card with the three of clubs. Display the new state of the object, and then repeat the test you performed in Step #7, and make sure you get the opposite outcome. Output from this step should look like this:

```
Replaced the 2 of diamonds with the 3 of clubs:
2 of clubs
3 of clubs
2 of hearts
2 of spades
Not four of a kind
```

Flush

9. Write the code for the `flush` method in the `PokerHand` class.

10. In the `testFlush` method in `Lab7App`, construct a `PokerHand` object consisting of the two of clubs, three of clubs, two of hearts, and two of spades. Display the line "`Testing flush method`" followed by the return value from the new `Hand` object's `toString` method.

11. Call the `PokerHand` object's `flush` method, and use an `if` statement to display either "`Flush`" or "`Not a flush`".

12. Replace card 3 with the four of clubs, and then replace card 4 with the 5 of clubs. Re-display the `PokerHand`'s state, and then perform the test from Step #11 again. Output from the `testFlush` method should look like this. ***Please note*** that the line that starts with "Replaced" ends with "5 of clubs", but that was too long to display on one line in a PDF.

```
Testing flush method:

2 of clubs
3 of clubs
2 of hearts
2 of spades
Not a flush

Replaced the 2 of hearts with the 4 of clubs and
the 2 of spades with the 5 of clubs:
2 of clubs
3 of clubs
4 of clubs
5 of clubs
Flush
```

Three of a Kind

13. Now that you know what you're doing, write the `threeOfAKind` method in the `PokerHand` class. Test it in the `testThreeOfAKind` method in the `Lab7App` class so that it generates this output:

```
Testing threeOfAKind method:

2 of clubs
3 of clubs
4 of clubs
5 of clubs
```

```
Not three of a kind

2 of clubs
2 of diamonds
2 of hearts
5 of clubs
Three of a kind

2 of clubs
2 of diamonds
3 of clubs
2 of hearts
Three of a kind

2 of clubs
3 of clubs
2 of diamonds
2 of hearts
Three of a kind

3 of clubs
2 of clubs
2 of diamonds
2 of hearts
Three of a kind
```

Pair

14. Write the `PokerHand` class' `pair` method, and test it in the `testPair` method in the `Lab7App` class. There are six different conditions that result in having a pair, so make sure your `pair` method reflects that. Test the `pair` method in the `testPair` method to generate this output. Notice that there are seven different tests – six that return true, and one that does not.

```
Testing pair method:

2 of clubs
3 of clubs
4 of clubs
5 of clubs
Not a pair

2 of clubs
2 of diamonds
4 of clubs
```

```
5 of clubs
Pair

2 of clubs
4 of clubs
2 of diamonds
5 of clubs
Pair

2 of clubs
4 of clubs
5 of clubs
2 of diamonds
Pair

5 of clubs
2 of clubs
2 of diamonds
4 of clubs
Pair

5 of clubs
2 of clubs
4 of clubs
2 of diamonds
Pair

5 of clubs
4 of clubs
2 of clubs
2 of diamonds
Pair
```

Low Card

15. Finally, write the code for the `lowCard` method in the `PokerHand` class, and test it in the `testLowCard` method. Your tests should generate this output:

```
Testing lowCard method:

2 of clubs
3 of clubs
4 of clubs
5 of clubs
The low card is 2 of clubs
```

```
3 of clubs
2 of clubs
4 of clubs
5 of clubs
The low card is 2 of clubs

3 of clubs
4 of clubs
2 of clubs
5 of clubs
The low card is 2 of clubs

3 of clubs
4 of clubs
5 of clubs
2 of clubs
The low card is 2 of clubs

3 of clubs
2 of diamonds
5 of clubs
2 of clubs
The low card is 2 of diamonds
```

As stated before, all of the expected output is listed in one file on the Lab Assignment's web page.

Submission

Upload `PokerHand.java` and `Lab7App.java` to Web-CAT and to my web site. On my web site, type either the name(s) of your partner(s), or a message indicating that you didn't have any. All files must be submitted before the start of Thursday's class.