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

Project #2: Pairs
Out: Tuesday, 31 May 2016
Due: Friday, 3 June 2016 11:59:00 pm

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

Please note that you are meant to work on this assignment yourself.

1 Objectives

- To understand and develop program logic
- To test assignment statementes, object instantiations, method calls, and output statements
- To add code to an existing program
- To design test cases to ensure a program works properly

2 Overview

In this project, you will add code to an existing project in order to add functionality. In the process, you will learn how to ensure that your code is written properly. Finally, you will take a short quiz in class on the due date to reinforce the important concepts.

3 Specifications

3.1 Introductory Steps

Download the files `Project2.java` and `cardgames.jar` from the project web site. Create a new Java project, add the code in `Project2.java`, and add the JAR file to the project. Re-read the instructions from Lab Assignment 3 if you need a refresher on how to perform these tasks.

If you run the code that has been provided, you will see two cards in the GUI, and some information about them in the console. Notice that because the `Deck` is shuffled each time you run the program, then the cards and the information about them will change each time you run it.

In your project report, state what cards and what information were displayed in five different runs of the program.

3.2 A Third Card

You must add statements to `Project2.java` to:

- Deal another card, and store it in a variable
- Display the information about that card in the console
- Add the new card to the GUI

Your console output should now be something like:

```
Card 1 is: 5 of hearts
Card 2 is: 12 of spades
Card 3 is: 6 of diamonds
Not a pair
```

Notice that the currently-existing code that checks whether the hand contains a pair is insufficient. Add code so that the program reports the existence of a pair *among any two of the three cards*. Make sure your program appears to be running properly. You might find it difficult to test extensively, since it's hard to predict whether a pair will be dealt or not; we will address that concern later in the project.

3.3 Point Totals

Add statements to determine and display the highest value of the three cards. Then add code to calculate and display the sum of the **point values** of the three cards. (Refer to the documentation to make sure you're calling the right methods.) If the sum of the point values is 25 or higher, display a message that the player wins; otherwise, display a message that the computer wins.

At this point, your program's complete console output should look like this:

```
Card 1 is: 7 of diamonds
Card 2 is: 12 of clubs
Card 3 is: 6 of hearts
Not a pair
The highest value is 12
The sum is 23
The computer wins!
```

3.4 Dealing with Randomization

You might have found it difficult to test this code as thoroughly as you would like, since you can't predict whether a pair will be dealt or not. You might not have encountered a pair yet; or perhaps you haven't gotten 25 points yet. If either of these are true, then you can't really know if your code is running correctly.

To convince yourself that your logic is correct, you need to be sure of what values are being tested. Find the code that deals cards from the deck; comment these lines out by adding `//` at the beginning, and also add your own comment above each of these lines describing what you've done and why. Then, below each line, write additional code to call the `Card` class' constructor directly. This constructor requires one `int` parameter, which determines the suit and the value of the card that gets instantiated. Table 1 contains some possible parameters, and the cards that are instantiated when these parameters are used. For instance, the line `Card card1 = new Card( 31 );` will instantiate the five of hearts.

Parameter	Card	Parameter	Card
5	5 of clubs	11	11 of clubs
18	5 of diamonds	24	11 of diamonds
31	5 of hearts	37	11 of hearts
44	5 of spades	50	11 of spades

Table 1: `Card` constructor parameters and results

In your project report, state which cards you used to test the various things that must be tested. You need to test a few different conditions to see if there's a pair of cards. You need to test a few different conditions to make sure your highest card code is working. And you need to do a few different things to make sure the over/under 25 code works properly. Discuss all this thoroughly in your project report.

3.5 Comments

Make sure to add Javadoc comments at the top of `Project2.java` including the title of the project, a description of what the code does, when it's due, and who you are. Add comments as you see fit throughout the code to explain what you've done.

4 Project Report

Create a ***plain-text*** file called `project2.txt` including the statements indicated above, as well as answers to the following questions:

1. How long did it take you to complete this project?
2. What kind of help did you receive on this project, and from whom?
3. What was the most difficult challenge you faced in this project?
4. Are you sure your program works? Why or why not?

Your responses to these questions will be graded based upon grammar and spelling as well as on content. Please use complete sentences. Any file other than a plain-text file will be deleted without ever being read or graded.

5 Submitting Your Project

You must submit your project using the interface on my web site:

http://www.matcmp.ncc.edu/~cmerlo/

No other submissions will be accepted. You may re-submit this project as often as you want; I will only ever see the most recent submission. Notice this means that if you submit before the deadline, and then re-submit after the deadline, your project will be late, and you will lose lateness points. You must submit the following files:

- `Project2.java`
- `project2.txt`

6 Assessment

It is clearly important that your program runs successfully. That is why **30%** of your grade will be based on the **correctness** of your program.

However, it is also important to write code that conforms to the rules we programmers have imposed upon ourselves, to ensure readability. **Ten percent** of your grade will be based, therefore, on your adherence to the **style guidelines** listed on my web site.

Additionally, **proper testing and proper analysis** are the tools we use to convince non-programmers that our code works. Therefore, **ten percent** of your grade will be based on your analysis.

Finally, you will take a short quiz at the end of class on the day of the deadline. The answers to this quiz will account for **50%** of your grade. You will be allowed to refer to a printout of your code during the quiz. This printout ***must*** be of your submission for `Project1.java`, and it ***must*** contain your name in the comments. It is ***strongly*** suggested that this printout include line numbers.

7 Things You Should Know

- As with Project 1, you may help each other, but **you are not permitted to share code.** Not even one line. If you're going to talk about this project with each other, leave the conversation empty-handed, and then write the code on your own. Submitting another student's code, or allowing another student to submit yours, will automatically earn both of you a zero on this assignment. Remember that earning a zero due to academic dishonesty also disqualifies you from withdrawing from the class.

- This project is due at 11:59:00 pm on June 3rd. Late projects lose 10 points per 24 hour time period or portion thereof, starting at 11:59:01 on June 3rd, regardless of weekends, holidays, weather, computer malfunction, etc. Any submissions uploaded after 11:59:00 pm on Monday, June 6th will be ignored, and a grade of 0 will be recorded.
- Store your data in multiple places. Consider using a system like Dropbox (https://db.tt/KEV2lqS) or SpiderOak (http://www.spideroak.com/). (If you decide to use Dropbox, let me invite you; we both earn an extra 0.25 GB of free space.)