CSIT 111: Fundamentals of Programming I

Project: Histogram Application

Due date: 12/17/2017, 23:59 PM (No Extension Allowed)

Instructions:

All submission must be digital form in Canvas. You have to submit a lab report and all the java source files. The lab report should include the lab number, your name, and the answers of all the questions. Please zip all files into one packet and name it with the course, section and lab number, and your name. For example: ***csit111_02_project_jane***.

Check List:

In this lab, you need to submit the following files:

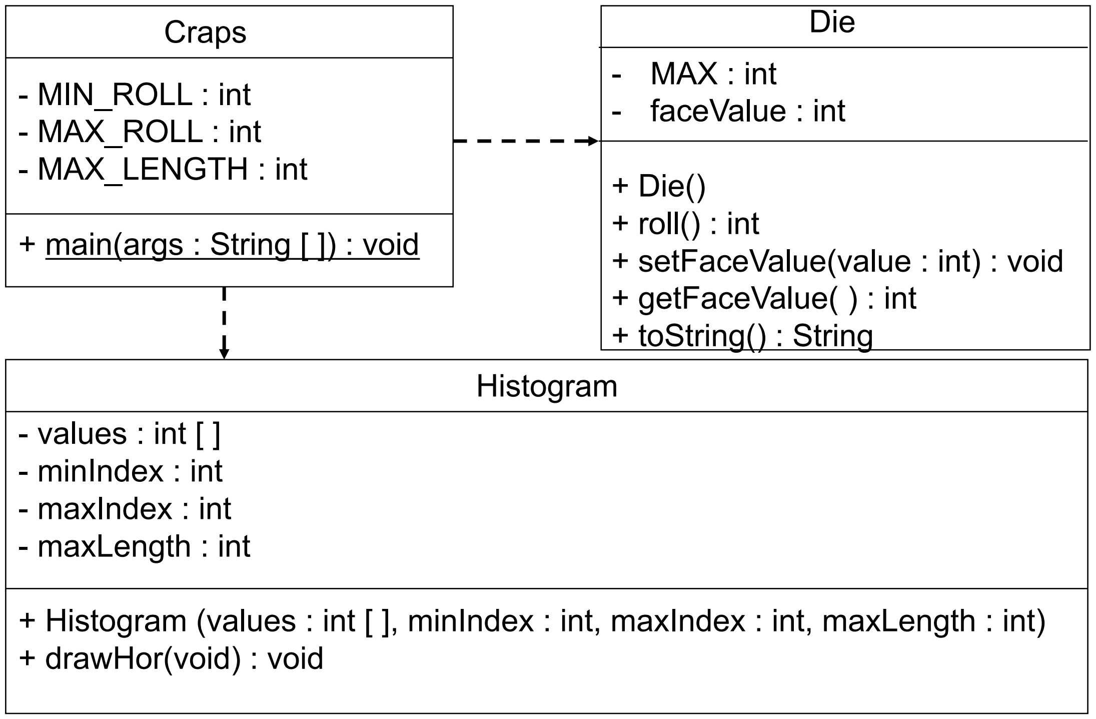
- ***Histogram.java***
- ***Craps.java***
- ***Die.java***
- ***Lab Report***

In a casino, there is a dice game called Craps. In Craps, you as the "roller" bet some money and throw two dice. If you roll a 7 or 11 on the first roll, you win. If you roll 2, 3, or 12, on the first roll, you lose. If you roll anything else, that value becomes your "point". Then, you keep rolling until you either: roll the value of your "point" again (and win) or roll a value of seven (and lose). This simple game is based on the probabilities for the values of the sum of two dice rolls. However, we are not going to play Craps. We will do an analysis of the frequency of occurrence of the total value of two dice rolls that may be helpful in understanding the probabilities if you ever do play Craps.

We are going to generate a histogram showing the frequency of occurrence for the sum of two dice rolls. We will display the histogram in the form of a horizontal bar graphs. This project will require you to write several loops that process the integer variables in an array.

You will use the Lewis and Loftus code for the **Die** class. You will not be modifying this code. You should read it and understand how it works. You will write code in the main method of a class named **Craps** and in a method of a class named **Histogram**. Here is a UML class diagram that explains the relationship between these classes:

Project UML Diagram



You can find an initial version of the ***Craps.java***, ***Histogram.java***, and ***Die.java*** source files in **Project.zip**. Download and unzip this file on your removable media and create/save a Dr Java project file in the Project directory. The ***Craps.java*** and ***Histogram.java*** file are incomplete and you need to complete them in this project. The ***Die.java*** file is complete and you do not need to modify it.

The code for the Craps class has the following constants pertaining to the game itself:

```
private static final int MIN_ROLL = 2;          // minimum value of a roll
private static final int MAX_ROLL = 12;         // maximum value of a roll
```

The code for the craps class also includes the constant for the maximum bar length in the histograms.

```
private static final int MAX_LENGTH = 36;    // maximum length of bars
```

These constants define the minimum roll value of 2, the maximum roll value of 12, and the maximum length of the bars in the bar graph.

The code that you add to the **Craps** main method needs to do the following steps in the following order (where indicated by the comments in the code itself):

1. Declare an integer array named "**counts**" with a size equal to MAX_ROLL + 1. Each integer element in this array will contain the count of times that a particular dice roll has occurred. For example, if 4 was the value of the dice roll 26 times, the array element "counts[4]" will contain the integer value 26. The lowest element in the array that you will use is "counts[MIN_ROLL]" and the highest element in the array that you will use is "counts[MAX_ROLL]". We will be ignoring the presence of "counts[0]" and "counts[1]" in the array because the roll of two dice can never total to either 0 or 1.

2. Execute a loop to initialize the value of all elements in the array from MIN_ROLL up through MAX_ROLL to 0. If we are going to start counting the number of times a particular value is rolled, we want to start all of our counts at zero.

3. After the code has read the value **n** (the desired number of rolls) from the user, execute a loop **n** times. Each time through the loop, your code calls the roll method of ***myDie1*** and ***myDie2***, adds the two values together, uses that sum as an array index, and increments the value of that element in the **counts** array.

4. Now calculate an estimate of the probabilities of the three Craps outcomes on the first roll (win, lose, or roll again). The probability of occurrence of an event can be estimated from its frequency. The estimate is better when there are a large number of samples in the run producing the frequency of occurrence data, e.g. >= 10,000. You can add the count values for 7 and 11 to get the number of occurrences of winning and divide by the total number of rolls to calculate an estimate of the probability of winning on the first roll. Likewise, you can calculate an estimate of the probability of losing and the probability of needing to roll again. Write code in your program to calculate and print these three probability values as shown in the sample output. Note that your probability values may be slightly different than the values shown in the sample output due to the random nature of the simulated dice roll process.

Be sure to cast your data from integer to **float** to perform this calculation since the probability values will all be between 0 and 1. (If you do the division in integer arithmetic, all answers will come out to be 0 due to the truncation of the fractional parts.)

It's nice to get estimates of these three probabilities, but it would also be nice to display the data to allow a user to observe the pattern of the frequencies for the various numbers of rolls.

5. At this point, your Craps code instantiates an object of Histogram class with the array of counts, the limits of the indices to draw, and the maximum length of the bars desired. It then calls the Histogram object's two draw methods.

You must write the rest of the code in the Histogram constructor method and its draw method. Note that this class has nothing to do with dice or the game of craps. The Histogram object's draw methods can draw histograms for any kind of data values that are passed to its constructor via its parameter list. Hence, we use "neutral" names for all variables in the Histogram draw method – not names like "rolls" that imply any such activity related to the game of Craps. The histogram methods need to do the following steps in the following order (where indicated by the comments in the code itself):

In the Histogram constructor method, we need to initialize the instance variables from the supplied parameters. This code has been provided for you. You should study it and what it is doing to understand it.

6. and 7. We may have very large values in the values array supplied as a parameter. Since we want to limit the size of the histogram bar graph to **maxLength**, we need to scale the data in the values array when we copy it into the instance copy of the data in the array. This consists of two loops. The first loop finds the largest value in the values array. (Declare, initialize, and use a variable named something like "**maxValue**"). The second loop multiplies each value in the values array by the max length we want for bars (**maxLength**) and divides by the largest value found ("**maxValue**").

8. In the ***drawHor*** method, your code must draw a horizontal bar graph of the data in the values array. (See Sample Output) This will require two nested loops. The outer loop will be a scan through each element of the values array and the inner loop will print an asterisk from 1 through the value in the outer loop element of the values array. Print the values array integer at the end of each bar of asterisks.

Sample Output:

```
How many dice rolls do you want? 1000

Estimated probabilities for outcome of the first roll:
Win:          0.214
Lose:         0.119
Roll again:   0.667



Value 2:  ******* 7
Value 3:  ********** 10
Value 4:  ************* 13
Value 5:  ************************** 26
Value 6:  ****************************** 30
Value 7:  ************************************ 36
Value 8:  *************************** 27
Value 9:  *********************** 23
Value 10: ******************* 19
Value 11: ********* 9
Value 12: ****** 6
```

<u>Lab Report</u>

Write a report that answers the following questions:

1. When we declared the **counts** array why did we need to add one to MAX_ROLL for the size of the array? What happens if you remove the + 1 from the declaration of the counts array size and run the program?

2. Each time you run the program with a large number of dice rolls (e.g. >=1000), you get a distinctive distribution and the distribution shows very similar values from one run to the next. What happens when you run the program several different times with a small number of dice rolls (e.g. <= 100 or <= 10)? Can you explain what is happening?

3. Why do we not want to use any names based on the game of Craps in the Histogram class code?

4. In the Histogram constructor, we made a scaled copy of each element of the data supplied in the original parameter array. Why did we do that and what would be the consequences if we did not?