Informatics 1: Object Oriented Programming

Assignment 2 - Report

<S2084333>
<March 5th 2021>

Basic

This section looks at the details of the implementation, lists out the features, a specification of the program, and some additional features that might be useful.

<u>List of desired program features:</u>

- The program takes 2 set of values (maximum and minimum of rainfall each month), and print the following constructed strings on the terminal:
  - Horizontal Monthly Precipitation Histogram (days on y-axis)
  - Horizontal Yearly Precipitation Histogram (months on y-axis)
  - Vertical Monthly Precipitation Histogram (days on x-axis)
  - Vertical Yearly Precipitation Histogram (months on x-axis)

- The program will process the given values in the following ways:
  - Calculate/generate an int value for rainfall on a specific day. The precipitation should be within the given month's range.
  - Calculate/generate a specific int value for rainfall in a month within the given range.

- A user given float value as the multiplier of the calculated rainfall values which scales the output.

- Each constructed string should include:
  - Type of the histogram (yearly/monthly)
  - The month if it's a monthly graph
  - Scale of the histogram (float value)
  - Histogram (months/days versus precipitation)

<u>Program specification:</u>

- The program uses a provided set of maximum and minimum rainfall precipitation data, and produces the following graphs:
  - Horizontal Monthly Precipitation Histogram
  - Horizontal Yearly Precipitation Histogram
  - Vertical Monthly Precipitation Histogram
  - Vertical Yearly Precipitation Histogram

  If it's monthly graph, each day is represented, if it's yearly graph, each month is represented.

- Each produced graph should be able to scale up or down itself based on user inputs.

- The produced graph should display the following:
  - Whether it's a yearly or a monthly histogram and the month if it's a monthly graph
  - Scale of the histogram
  - Histogram itself

- The amount of rainfall on a specific day and the specific amount of rainfall in a given month are not given. Generate suitable values in the program. Total amount of rainfall in a month shouldn't be above or below the limits.

Additional requirements

- There should be some TUI features that ask for user inputs, for example which graph would the user like to print out or the scaling multiplier.
- Allow user to set the maximum and minimum rainfall as well as the year, from which whether it's a leap year and the number of days in February can determined.

Intermediate

This section lists the errors and issues with the code and possible solutions or improvements to the listed problems.

Program functionality		
Requirement	Issue	Proposed solution
Monthly plot should display a certain amount of rain for each day in the month.	There are always only 12 days being displayed.	The error is in line 66 in PrecipitationGraph.java, where the month argument and day argument are reversed. Switch their order to resolve this issue.
Monthly plot should display 28 days for February as given by the dataProvider.java file.	There are 29 days being displayed with the 29th day being empty.	The error is in line 38 in PrecipitationGraph.java, where 29 days are supplied instead of 28. Change the number of days to 28 to resolve this issue.
Each monthly precipitation should be above the minimum rainfall value.	Multiple months average rainfall precipitation value are below the minimum.	The error is in line 66 in PrecipitationGraph.java, where the month argument and day argument are reversed. Switch their order to resolve this issue.
The precipitation for January should be displayed according to Test.java	February is displayed instead.	Both the String[] monthName in line 18 and the int[][] rainfall in line 57 in PrecipitationGraph.java are 0 indexed. By subtracting 1 from the input in monthVertical in line 228 and monthHorizontal in line 210 in the same file mentioned above to resolve this issue.
Each daily precipitation should be above the minimum rainfall value.	Lots of daily precipitation is below the minimum.	The error is in line 65 in dataProvider.java, the wrong function is used. Change minRainInMM to maxRainInMM to resolve this issue.
Each daily precipitation should be below the maximum rainfall value.	Occasionally daily precipitation is above the maximum.	The error occurs from line 65 to line 69 in dataProvider.java as maxRainToday has the possibility to exceeds the maxRainThisMonth (assuming the logical error in line 65 has been fixed). It can be resolved by using the maxRainThisMonth as maxRainToday, or adding an additional check if the maxRainToday has exceeded maxRainThisMonth similar to line 72 to 74.

Code quality			
Code affected	Issue	Proposed solution	Explanation/justification
Line 79 in PrecipitationGraph.java	Structure: Casting to int is not necessary as it was casted again in line 129.	Remove the redundant casting in line 79.	Reduce code duplication.
3 fields in class dataProvider. (Line 14, 19, 44)	Robustness: Access level modifiers.	Change minRainInMM, maxRainInMM, and rand into private fields.	Restrict exposure in class PrecipitationGraph.
All fields in class dataProvider. (Line 14, 19, 29, 44)	Robustness: Access level modifiers.	Remove “static” modifiers from minRainInMM, maxRainInMM, daysPerMonth, and rand.	Restrict exposure in general.
Field “DAY_PER_MONTH” in class PrecipitationGraph	Structure: The number of days in each month was listed twice	Set this field to: DAYS_PER_MONTH = dataProvider.daysPerMonth	Reduce code duplication.

Method “findMax” in class PrecipitationGraph	Technical proficiency: The method is not efficient and hard to read.	Use java.util.stream:
		int max = Arrays.stream(array).max().getAsInt();
		return max;
			Code simplification, a cleaner solution.

Method “preparedData(int[] month)” in class PrecipitationGraph	Technical proficiency: There is a shorter and cleaner solution.	Use java.util.stream to replace the for loop: return Arrays.stream(array).map(x -> (int)(scale * x)).toArray();	Code simplification, a cleaner solution.
From line 187 to line 207 in PrecipitationGraph.java	Structure: Modularity and reusability.	Extract line 187 to 207 as a separate private method.	Modularize overly long and specific method.
Line 171 in PrecipitationGraph.java	Technical proficiency: for loop replaceable with for each loop.	Replace for loop with a for each loop: for(int entry : array){}	Code simplification, a simpler solution.
Method “monthVertical” and “monthHorizontal” in class PrecipitationGraph	Robustness: Runtime error occurred while attempting to print month name.	Subtract 1 to the input “month” in both methods, as field “monthName” is 0 indexed. Also add in a try-catch block or a separate method for checking input range to avoid index out of range exceptions.	Avoid crashing.
From line 63 to line 69 in dataProvider.java	Structure: Modularity and reusability	Extract them into a separate private method.	Modularize overly long and specific method.

Note: If you want to refer to code by line number, you must use the line numbers in the provided code. These may change if you e.g. add comments to the code to help your own understanding!

Code documentation			
Code affected	Issue	Proposed solution	Explanation/justification
Line 63 to 69 in dataProvider.java	No clarification on the purpose of calculating maxRainfallDifference.	Comment a basic description of the purpose for this calculation.	Explaining how is daily rainfall determined makes maintaining your code simpler and making sense of your code reduces error.
Line 124 to line 134 in PrecipitationGraph.java	No documentation on the method overloading.	Add a short description on how overloaded methods are called and their purpose.	Explaining how to call your methods properly makes it easier for others to use your code and reduces error.
Line 119 in method "setScale"	Uninformative and needless comments.	Remove useless comments.	Having unnecessary comments clutters the code, making it hard to read and causes confusion.
All comments in method "verticalGraph"	Needless comments.	Remove these comments.	Having obvious and unnecessary comments clutters and obscure the code.
All comments in method "findMax"	Distracting and inappropriate comments.	Replace them with a short note, for example, "require rework".	Having distracting comments makes the code hard to read and misleading.
Line 14 in PrecipitationGraph.java	Misleading and incorrect comments. Field rainfall is indexed as [month][day] instead of [month][year].	Replace "indexed as [month][year]" to "indexed as [month][day]".	Comment variables incorrectly creates confusion for readers and potentially causes logical errors in future coding.

Advanced

This section looks at a piece of legacy code, describes the details and purpose of it, and provides suggestions about more readable code. Questions about the data structure of the legacy code is also answered.

Method "m1":
Proposed method name: <TODO>
Description of what it does:

- Double x by signed left bitshifting 1 bit
- If the given Boolean is true, increment x by 1
- Increment y by 1

Description of how it works: <TODO>

Method "m2":
Proposed method name: ifXOdd
Description of what it does:

- If y equals to 0, return null
- Otherwise return whether if x is odd.

Description of how it works: <TODO>

Method "m3":
Proposed method name: <TODO>

Description of what it does:

- Save the result of m2 to a variable called “z”
- If z isn’t null, divide x by 2 using unsigned right bitshifting by 1 bit, then decrement y by 1
- Return z

Description of how it works: <TODO>

Method “m4”:
Proposed method name: setXYZero
Description of what it does:

- Set both x and y to 0

Description of how it works: <TODO>

Method “m5”:
Proposed method name: getY
Description of what it does:

- Return y

Method “m6”:
Proposed method name: ifYEqualsZero
Description of what it does:

- Return whether if y equals 0

Question 1: What kind of data structure is this and what would be a better class name?
<TODO>

Question 2: What are the advantages and disadvantages of the chosen data representation?
Advantages:

- The brevity of this piece of code will reduce memory signature.
- The code is reusable in different circumstances because of its abstract nature.
- The code can be easily tested.

Disadvantages:

- The readability of the code is not well.
- The code will be difficult for future maintenance.

Question 3: Is there any justification for writing code like this (why/why not)?
The code focuses on performance and efficiency, and aims for minimum memory signature. This is justified for usages where the available memory is really scarce. (e.g. a space probe mission or a hyper accurate simulation program, where every bit of memory counts.) For daily usages, there are normally sufficient spare memory for programs to be more readable, with better user interfaces, and makes them much easier to maintain.