Informatics 1: Object Oriented Programming

Assignment 2 - Worksheet

S2029893

Basic: Technical Description (Step 5)

public int getRain(int month, int day)
This method helps us to generate a random amount of rainfall for the date we asked. The method takes two integer inputs (month and day) and returns an integer output (rain). It first checks if our inputs are valid or not by checking the conditions of whether the input value is over or under the valid month value, returns -1 if inputs are invalid. Next, it defines and initializes the minimum and maximum amount of rainfall by getting the value of corresponding month from the public array which contains the final value of minimum or maximum amount of rainfall. After that, it adds the offset to the maximum rainfall of the day we asked by calculating the distance through that day to the next month. Finally, it returns the randomly generated amount of rainfall in the calculated range.

PrecipitationGraph()
This method initialises the rainfall array and fills the array with values of amount of rainfall for every day of the year. It first initialises the number of columns of rainfall array with the number of months which is 12. Next, by iterating through all months in a for loop, it initialises the number of rows for each column with the number of days for each month. So far the PrecipitationGraph() method has initialised the rainfall array which has 12 months and each month has a different month lengths according to the number of days it has. Finally, by iterating through all days in a for loop within the first for loop, it accesses the value of amount of rainfall for each days using the getRain() method we built in class dataProvider.

private int[] prepareData()
This method helps us to generate an array with average amount of rainfall of months in one year. It first defines and intialises an array with length 12. Next, by iterating through all months in a for loop, it calculates the scaled monthly average amount of rainfall by multiplying the float number scale with the monthly average amount of rainfall which was accessed by calling monthlyAverage() method for each month.

private void verticalGraph(int[] array)
This method helps us to generate a graph contains the details of amount of rainfall with corresponding months or days in a vertical version (axis of months or days is lying horizontally at the last row). This method takes one input (an integer array) and returns nothing but prints a graph. It uses a String named stars to store stars (*) which represents the amount of rainfall. For the graph printing part, it first prints each row of stars representing the amount of rainfall. By iterating through the value from the largest entry in the given array to zero with a for loop, it can print the stars from the highest amount of rainfall to zero like a histogram, in this case, vertically. Within the first for loop, it uses another for loop which iterating through the length of inputted array to see whether the value of array for the current row should print a star (if the value is >= the current line height, it adds a star to the String stars; If not, it adds a blank). Next, it creates a dotted line (for separating the stars and months or dates) and date line (display the dates or months) by iterating through the inputted array's length with a for loop. For every entry it makes the dotted line longer by adding "---" to the String dottedLine and adds the number to the String dateline and finally after the for loop, it prints them.

Basic: Program specification (Step 6)

This program is intended to give an implementation of printing a precipitation graph of amount of rainfall. The rainfall data used to print the graph should be synthetically generated in the program. Also, this program should be able to print both yearly precipitation and monthly precipitation (for the month we choose). In addition, it should allow us to choose the display mode of the graph (both horizontal and vertical) and set the scaling value to scale the printed graph.

Basic: Additional features (Step 7)

Adjustable year

This program should be able to synthetically generate the rainfall data for different years, not only for data of months in the same year. This allows users to select the year of rainfall they are interested.

Visible corresponding value of amount of rain (in mm)

This program should be able to print the value of amount of rainfall in the graph instead of just showing the stars to the users. This can help users to know the exact values of the amount of rainfall easily when looking at the generated graph.

Intermediate: Code Review

Functional correctness		
Requirement	Issue	Proposed solution
The program should be able to calculate the maximum rainfall difference by subtracting the maximum rainfall for next month and maximum rainfall for this month.	Instead of subtracting the maximum rainfall for next month and maximum rainfall for this month, it subtracts the minimum rainfall for next month and maximum rainfall for this month.	The error occurs on line 65 in class dataProvider, where you should change minRainInMM[(month + 1) % 12] to maxRainInMM[(month + 1) % 12].
The program should be able to calculate the rounded scaled value after users set the scale.	Instead of rounds the data to the nearest integer, it only takes the integer part of the float number.	The error occurs on both line 129 and line 139, where you should use math.round() method to round the float value of before (scale * monthlyAverage(month)) and (scale * month[day]) instead of (int).
The program should be able to generate the precipitation for the month which users selected.	Instead of generates the precipitation for the month which users selected, it generates the precipitation for the month after the month which users selected.	The error occurs on line 213, 216, 231, and 234, where the inputted value of months (there are four of them) inside the monthName and rainfall arrays should be subtracted by 1, otherwise these two methods will generate the graph for another month.

Code quality			
Type of issue	Code affected	Issue and proposed solution	Explanation/justification
Naming conventions	Line 9 in class DataProvider	Issue: The class name is not written in PascalCase. Solution: Change the class name from dataProvider to DataProvider.	The class name should be written in PascalCase. That means that every word starts with a capital letter and all other characters are lower case.
Descriptive naming	Line 92 in class PrecipitationGraph	Issue: The simple accessor method findMax() is not named in setSomething(...) form. Solution: Chenge the method name from findMax() to getMax().	Simple accessor methods (aka. “getters”) should be named: getSomething().
Consistent formatting	Method getRain() of class DataProvider	Issue: The style for opening and closing braces used in this method is different to the rest of the program. Solution: Align opening and closing braces vertically.	Consistent formatting throughout the entire project is essential. It’s hard for readers to read if there are two different styles for opening and closing braces. More seriously, it can lead to misunderstanding.
Access modifiers	Line 54 in class PrecipitationGraph	Issue: There is no access modifier specified to the method precipitationGraph(). Solution: Add public to the method precipitationGraph().	By using restrictive access modifiers, we can prevent incorrect usage in the first place. If we do not specify an access modifier to this method, it will access default to “package private” which is not what we want in this case.
Java proficiency	Method getRain() of class DataProvider	Issue: this method uses too many extra spaces to define and initialise some useless variables. Solution: Calculate the final value we are looking for (rain) directly by accessing the value from the corresponding arrays (minRainInMM and maxRainInMM).	We should consider time and space complexity when we are coding. Define and initialise variables which we only just use for one time is a waste of space.
Code duplication	Array DAYS_PER_MONTH of class PercipitationGraph	Issue: This array defined and initialised in the class PercipitationGraph is the same as the array daysPerMonth in class DataProvider. Solution: Delete this array and access the array in class DataProvider in the future codes when needed.	Duplicated code makes the program longer than it needs to be and makes it difficult to maintain consistency.
Validation	Method monthHorizontal() of	Issue: The program crashes when the value passed as a parameter is out of the range of months.	If the program crashes, the user must start over. It is more user friendly to display an insightful error message and allow for

	class PrecipitationGraph	Solution: Test for and handle invalid input by using if statement. If the input is out of the range of months, print this error using System.err.println() and ask the user to correct their input.	invalid input to be corrected. By catching the error, you can avoid the program from crashing and assure the output is valid.
Modularity	Method verticalGraph() of class PrecipitationGraph	Issue: This method uses 52 lines to implement, which is too long. Solution: Remove the implementation of printing stars and lines into another independent method.	Methods should perform well defined tasks. A lack of modularity makes it hard to extend a program, because adding new features requires the specific methods to be completely rewritten.
Structure	The whole of class PrecipitationGraph	Issue: This class contains both data providing and graph printing tasks which make the responsibilities of this class vague. Solution: Remove some methods and arrays which take the responsibilities of providing data (like rainfall, monthName, monthlyAverage()…) into class DataProvider.	Programs should be designed to structurally divide responsibilities. A good design should split the task into more manageable chunks by a meaningful division of the code into classes. By doing this, you can make the structure of code clear and easy for readers to understand the code.
Missing comment #1	Line 6 of class PrecipitationGraph	Issue: This class is not commented. Solution: Class PrecipitationGraph need a comment at the top, explaining its general purpose.	Every class should have a comment at the top, explaining its (general) purpose. By commenting the class, you can make the readers easier to understand what this class is used for.
Missing comment #2	The whole of class PrecipitationGraph	Issue: All the methods are not commented at all. Solution: Every method should have a comment stating what it does. At the very least, methods monthlyAverage(), preparedData() and horizontalGraph() need an explanation of the parameters they take.	Explaining what your methods do will make readers understand your code easier. In addition, explaining how to call your methods properly makes it easier for others to use your code and reduces error.
Needless comment #1	Line 173, 176, and 180 of method verticalGraph() of class PrecipitationGraph	Issue: These comments explaining what the codes do are unnecessary. Solution: Remove those comments.	Those comments just re-state what the code is doing. The condition in the if statement is already very clear in the code and “+=” statement is very basic in Java. In this case, make comments on those codes will be pointless.
Needless comment #2	Line 200, 203, and 206 of method verticalGraph() of	Issue: These comments explaining what the codes do are unnecessary.	Those comments just re-state what the code is doing. The System.out.printin() is a very basic statement in Java which print the characters. In this case, make comments on those

	class PrecipitationGraph	Solution: Remove those comments.	codes will be pointless.
Misleading comment #1	Line 14 of class PrecipitationGraph	Issue: This comment gives the wrong description of the array indexes. The array was indexed as [month][day] not [month][year] Solution: Change the content after “indexed as” from “[month][year]” to “[month][day]”.	The comment will mislead readers since the actual index of rainfall should be month and day, not month and year.
Misleading comment #2	Line 94 – line 97 of method findMax() of class PrecipitationGraph	Issue: This comment is distracting. It uses too many words to explain the insights behind the implementation instead of how the code works. Solution: Remove this comment and replace it with the main point of the method.	The comments need to be not misleading and distracting and you need to avoid long, convoluted explanations of your thought process when writing the code. By replacing this comment with the main point of the method, you can let the readers know what this method is used for, not how you think about your code.

Advanced: Legacy code

Method “m1”:
Proposed method name: addBit
Description of what it does: Append a binary digit to binary string
Description of how it works: Shift the bit pattern of the integer number x to the left one position by using left shift operator, increase x by 1 if our input is true by using the if statement, and finally increase y by 1.

Method “m2”:
Proposed method name: isOdd
Description of what it does: Determine whether x is odd or not.
Description of how it works: First check whether is equal to 0 or not, if y is 0, return null; if y is not 0, it returns the Boolean value of “x % 2 != 0”, which returns true if x is odd and false if x is even.

Method “m3”:
Proposed method name: removeBit
Description of what it does: Remove last digit of the binary string
Description of how it works: First initialise a Boolean z by using the “m2” method, which can be used to check whether x is odd or not. Next, check whether the value of z is null or not, if z is not null, shift the bit pattern of the integer number x to the right one position by using the logical right shift operator and decrease y by 1. Finally return the value of z.

Method “m4”:
Proposed method name: reset
Description of what it does: Reset the binary string
Description of how it works: Set the value of x equals to y and y equals to 0.

Question 1: What kind of data structure is this i.e. what would be a better name for class X?
The data structure is binary string. The name for the class can be BinaryNumber.

Question 2: What are the advantages and disadvantages of the chosen data representation?
Advantage: Easier for us to read and implement the data. Disadvantage: Complicated to do the conversion.

Question 3: What is categorically wrong with X.java and why?
The code of the methods is not split into manageable chunks, also there is no comment at all.

Question 4: What aspect of the code can be justified and under which circumstances?
Both code quality and documentation can be justified to let others read and modify the code.

Advanced: Design patterns

MVC design pattern is stands for Model-View-Controller design pattern. Just like its name, the whole project with this pattern is separated into three components: Model, View, and Controller. Each component is responsible for a specific module of the project.

Model is the central component of the pattern, and it is responsible for managing the data of the program. In our first assignment, Model.java is implemented as the Model component of the pattern. It represents the state of the connect four games. Those method used to get data such as the new game board, current move, judgement of new move are implemented in this pattern.

View is the component output all those representations of information such as chart, diagram, or table. In our first assignment, TextView.java is implemented as the View component of the pattern. We implement all the display methods in this pattern so that the game can display the information such as welcome message, active player, and board to the users.

Controller is the component to accept user’s input and converts it to commands for the Model component and View component. In our first assignment, Controller.java is implemented as the Model component of the pattern. It controls the flow of the connect four games. This component basically set the structure and sequence of how our program will be implemented. For example, we implement all the methods which will be used to control the flow of the games in this component, such as display welcome message, display the active player, and let the active user chose to move, etc.

We can use this design pattern to separate our application's concerns, make our program more readable and logical. In addition, MVC pattern makes it easier for us to extend a program, because adding new features requires the specific methods to be completely rewritten and MVC pattern MVC design pattern brings more modularity which allow us to focus on a specific module of the program and we don't need to worry about other modules' functionality beside this module.

References:
[1] https://www.tutorialspoint.com/design_pattern/mvc_pattern.htm
[2] https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller#:~:text=Model%E2%80%93view%E2%80%93controller%20(MVC,and%20accepted%20from%20the%20user.