Informatics 1: Object Oriented Programming

Assignment 2 - Report

s2089135
5. 3. 2021

Basic

Overview:
This section covers the features available to the user of the program called Precipitation Graph, both in technical and prosaic manners. It also contains suggestions for extra features users could find helpful.

List of desired program features:

- The method monthHorizontal() lets user choose a month and displays precipitation data for every day in desired month in form of a bar graph. The bars are represented horizontally (along x-axis)
- The method monthVertical() does the same thing as monthHorizontal(), except that the bars are represented vertically (along y-axis)
- The method yearHorizontal() lets user display the average precipitation for each month in a year in form of a bar graph. The bars are represented horizontally
- The method yearVertical() does the same thing as yearHorizontal(), except that the bars are represented vertically
- The method setScale() lets the user choose a scaling for the representation of the precipitation data

Program specification:
This program should generate synthetic but somewhat realistic precipitation data for every day in the year based on the minimum and maximum precipitation recorded in a given month. Users should be able to display the precipitation data in form of bar graphs where bars are represented by stars and one star represents 1mm of rainfall. They should also be able to choose the scaling of the stars representing the data and choose the orientation of the graph – vertical or horizontal. Users should have the opportunity to choose if they want to show precipitation data for every day in a chosen month, or average precipitation for every month in a year.

Additional requirements
First feature I would add is a better legend for the graphs. The representation of rainfall by stars in graph is useful if the user just wants to take a glance and see when it rains the most/least, but if the user wants to know the exact amount of rainfall, he/she must count manually all the stars, and so by adding a legend which would show the amount would greatly improve readability for the users. Also having the option to represent the data in a table could improve readability.
Second feature I would add is more statistics. The program should let the user order the data by the most rainy/dry months or get the top ten most dry/rainy days in the year for example. Having access to all kinds of statistics and trends could help for example farmers to plan out when do they need to use more/less water and when is the ideal time to sow or plant.

Intermediate

This section reviews the code. Its aim is to spot all the errors and find a way how to fix them. This section will also give suggestions how to improve the structure, robustness and readability of the code.

Program functionality		
Requirement	Issue	Proposed solution
Methods monthHorizontal() and monthVertical() in class PrecipitationGraph should show data for every day in desired month	These methods show data only up to the 12th day – the rest is always empty	The error occurs in the constructor of class PrecipitationGraph on line 65 – when calling getRain() method of class dataProvider, first parameter inputted is day and second month, but the method expects it the other way around and so it should be swapped
Methods monthHorizontal() and monthVertical() in class PrecipitationGraph should show data for appropriate months based on input (1 = January, 2 = February etc.)	The months are shifted to the right – if user inputs 1, he gets data for February, if 12, he gets an error instead of December	Methods monthHorizontal() and monthVertical() in class PrecipitationGraph should subtract 1 from the parameter month, because both arrays used in these methods (monthName and rainfall) start with an index of 0
Data scaled by method setScale() in class PrecipitationGraph should be rounded to the nearest integer	Instead of rounding it to the nearest integer, the data is always rounded down	The issue lies on lines 79 and 129 in class PrecipitationGraph – instead of casting to (int) which always rounds down, Math.round() should be used to round doubles to the nearest integer
Method monthlyAverage() in class PrecipitationGraph should return the average precipitation for a month	It does not return the correct number for the month February	The error occurs in line 38 of class PrecipitaionGraph – February is given 29 days instead of the correct 28 days and so the average is always a bit lower than in reality
Method findMax() in class PrecipitationGraph should return the largest value from an inputted array	It does not take into account negative numbers (This does not apply specifically to this program, but in general this method does not do what it should correctly)	The issue lies on line 98 – if the array would contain only negative integers, the max value would be 0 – which is not contained in our array. The variable temp should be set to the first entry instead of 0
Method getRain() in class dataProvider should always generate rainfall data for certain day in a month in range of the months' max and min recorded rainfall	In some cases it can generate more rainfall than it should, for example: on the 28th of February, getRain() can generate 40mm of rain although the max recorded is 38mm. distanceThroughMonth = 28/28, maxRainToday = 38 + (2 * 28/28), and if rand.nextFloat() happens to be 1.0, then we get 40mm	Delete all of these calculations and just generate a random number between max and min recorded rainfall for given month. If the author would insist on generating the data this specific way, he/she could check if "rain" is more than the maximum rainfall for the month, in the same fashion as it already does for minimum rainfall
Variable maxRainNextMonth of class dataProvider should give us the maximum recorded rainfall for the next month	It gives us the minimum recorded rainfall for the next month instead of maximum	Instead of setting this variable to minRainInMM[(month+1) % 12] it should be maxRainInMM[…]

Code quality			
Code affected	Issue	Proposed solution	Explanation/justification
Method monthVertical and Horizontal in class PrecipitationGraph	Robustness: Program crashes when user input for desired month is not in range [0, 11]	Test if input is an integer and if it is in range of the number of months	The program will not crash anymore, user will be asked for valid input and will be told if input is not an integer or is not in appropriate range
Variables on lines 14 and 19 in class dataProvider, line 18 in class PrecipitationGraph	Robustness: These variables should use different access level modifiers	Change the access level modifier from public to private for lines 14 and 19. Line 18 in addition to private and final should be static	Variables on lines 14 and 19 are static constants used only in dataProvider to generate rainfall data and no other class or user should have the option to alter and have access to this data. Variable on line 18 in PrecipitationGraph contains data which should stay the same for all instances of this class, so it should be static
Method getRain() in class dataProvider	Structure: This method is very long and hard to read and understand	Keep getRain() as the "main" method in which the data is built from other smaller methods	There are too many variables cluttered on each other doing some calculations and it is very hard to understand it. Splitting these calculations into small methods would make the code more compact, readable and understandable
Method preparedData() in class PrecipitationGraph	Technical: Using a hard coded integer in a condition in for loop (line 127)	Instead of having "month < 12", better practice would be to have "month < year.length"	Even though the number of months is always 12 and it would not matter in this specific case, for reusability it is better to check for the length of the array we are looping to avoid errors
Methods preparedData() and preparedData(int[] month) in class PrecipitationGraph	Readability: Methods have exactly the same name even though they do different things	Change the names to be more self-explanatory and descriptive	Using a same name for two methods can be done for overloading, but in this case both of these methods do different things, and it is very misleading. Also, the name preparedData() does not tell the user/programmer anything about its functionality
Variables minRainInMM (class dataProvider), ArrayMax (class PrecipitationGraph), class dataProvider	Readability: Incorrect naming conventions for these examples (there are more)	As a final static variable, minRainInMM should be written in SHOUT_CASE, variable ArrayMax should be arrayMax and class dataProvider should start with capital D.	Naming conventions make it easier for other programmers to read your code. Not adhering to them can cause confusion
Method verticalGraph() in class PrecipitationData	Structure: Too many tasks in one unnecessarily long method	Lines 187 to 204 could be in a separate method which formats these dotted and date lines based on the inputted array	A method that formats these lines for vertical graphs could be used in more ways than just in this specific method. It would also make the method verticalGraph() more compact

Line 64 to 65 in class PrecipitationData and the whole dataProvider class	Readability: Inconsistent formatting	The loop on lines 64 and 65 is the only case where no curly brackets are used, the rest of PrecipitationData always opens curly brackets on the next line. DataProvider on the other hand opens curly brackets on the same line as the method/conditional/variable	Consistent formatting makes the code more readable
Method getRain() in class dataProvider	Technical: Using overly complicated way of doing something simple	For every day in a month, a random number between max and min rainfall should be generated	This method does not simulate reality in any way, because rainfall does not progress throughout the year in linear fashion based on max/min monthly rainfalls. Generating random number between min and max rainfall serves its main purpose – generate somewhat realistic data to test our graphs. The way it is coded now is neither realistic enough for precise simulation nor simple and compact enough for a mere random generator method

Code documentation			
Code affected	Issue	Proposed solution	Explanation/justification
Both preparedData() methods in class PrecipitationData	Code is not commented at all	At the very least, both of these methods should have a one-line comment which sums up what these methods do and what is their output	Completely uncommented code which in addition also uses vague names is very hard to read and understand
Method getRain() in class dataProvider	This method could use more comments	Write comments about the calculations which are happening inside	It is very confusing what exactly is going on in this method and it is hard to tell if some of the calculations are incorrect or if they were really meant to be implemented the way they are
Lines 21 to 28 in class dataProvider	Useless comment	Delete it	This poem may be a helpful tool for remembering the number of days in months, but it is not necessary to have it in our code, it only gets cluttered
Lines 17, 34, 113 and 148 in class PrecipitationGraph	Comments stating the obvious	Delete them	If the only thing a function does is “return scale;”, there is no need to comment that it returns the scale
Lines 94 to 97 in class PrecipitationGraph	Inappropriate, misleading and distracting comment	Either delete it as a whole or rewrite it so that it briefly sums up the code’s purpose	It clutters the code, it is written in spoken English which in itself is painful to read, and it misleads the reader that it does “massively complicated stuff”
Line 14 in class PrecipitationGraph	Misleading comment	Rainfall should be indexed by [month][day] and not [month][year]	[month][year] indexing is incorrect, it misleads the reader, and it could result in unnecessary errors

Advanced

This section inspects legacy code written by a retired programmer and tries to decipher it.

Method “m1”:
Proposed method name: push()
Description of what it does: First it checks it tries to shift bits to the left (multiply by 2^1). Nothing happens if x is 0, so the condition I assume checks if x is zero, and if it is true, it increments x to 1. Then it also increments y.
Description of how it works: X represents the stack and y represents its size. If both are 0, then the stack is empty, so we first add element and make x = 1 and then also y = 1. Now when x, y = 1, we can push in elements by shifting to the left -> x is now 2 (10 in bits) and the size y of stack is also 2. Another shift gives x = 4 (100 in bits) and size y = 3.

Method “m2”:
Proposed method name: check()
Description of what it does: First it checks if size y is 0, if not, it returns the state of the stack.
Description of how it works: If size y is 0, it means that the stack does not really exist and so the state of the stack is “null”. Otherwise, it checks if the number is odd, which comes in handy in the next method.

Method “m3”:
Proposed method name: pop()
Description of what it does: First it gets the state of the stack. If it is not empty, it shifts the bits to the right and fills out the space with zeroes. Then it decrements the size y. In the end it returns the state of the stack.
Description of how it works: If z is null, it means that stack is empty, and we can’t pop anything from it. If it is not, then we pop the last added element, decrement the size y of the stack and return the state. Here our method check() comes in handy, because if x is 1, it means that z is true and that means it has only one last element. Then we pop this last element, our stack is empty, and it returns true as in “it is true, that our stack is empty”.

Method “m4”:
Proposed method name: reset()
Description of what it does: It sets stack x and size of stack y to 0
Description of how it works: It resets the stack and its size to make it empty

Question 1: What kind of data structure is this and what would be a better class name?
This data structure is stack and better class would be StackOperations.

Question 2: What are the advantages and disadvantages of the chosen data representation?
The main advantage would be efficiency, because operating with bits is more natural for computers. Disadvantage is, that it is a bit less readable/intuitive for humans. You can also divide/multiply only by powers of 2, so it is not very flexible.

Question 3: Is there any justification for writing code like this (why/why not)?
It is better to write code this way if we are operating on more “deep” hardware levels, if we were for example manually managing the computer’s memory. It would not really make sense if we were creating an API for selling ice cream for example, unless all the prices, sales and accepted currency is somehow based on powers of two.