School of
Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

Assignment 3

Intro to Programming: Assignment 3

- Due Nov 10 23:30

Goals

This assignment will give you practice at writing methods with:
- methods that call methods in the same class
- methods that return values
- for loops

Resources and links

- Download Zip file of necessary code and data.
- Source Code and data Files (for viewing online)
- Submit your answers.
- Marks and Feedback (When available)

Summary

- Tricolour Flag Drawer:
  ➡ Write a program to easily draw flags made up of three stripes.
- Temperature Analyser:
  ➡ Write a program to analyse a series of temperature levels.

Overview

The assignment involves two programs to submit:
- The first program draws three-stripe flags
- The second program analyses a sequence of temperature levels, calculating some simple statistics and plotting them on a graph.

This assignment will contribute 2% of your final grade. Each part of the assignment will be marked out of 100, as for assignment 2. Your mark will be returned electronically, along with feedback comments from the markers.

To Submit

- Your version of `TricolourFlagDrawer.java`
- Your version of `TemperatureAnalyser.java`
- Your `Reflection.txt` file

Preparation

Download the zip file for assignment 3 and extract it to the `COMP102-2017T2-Assig3` folder in your home folder. It should contain templates for the programs you are to complete. Read through the whole assignment, and run the demos so that you know what you need to do.

Look at the model answers to assignment 2, and make sure you understand all the components of the programs.

Program 1: Tricolour Flag Drawer

In assignment 1 you created a program for drawing a variety of flags and symbols. In this assignment you will create a specialised flag drawer that allows you to easily generate flags that consist of three equal-size stripes (e: France, Italy, Russia, Germany, etc).

Core

- Complete the `drawThreeColourFlagCore` method to draw a three-colour flag consisting of three vertical equal-width stripes (like the French flag). You will need to specify the method's parameters as well as write the body. Note: the `doCore` method, which calls `drawThreeColourFlagCore` is already written for you.

Completion

- Complete the `drawThreeColourFlagCompletion` method to draw flags where the stripes can be either horizontal or vertical (so you can draw the Russian flag as well as the French and Italian). You will need one more parameter in your method than in the core. Note: the `doCompletion` method, which calls `drawThreeColourFlagCompletion` is already written for you.

Challenge

- Write a `doChallenge` method to use your `drawThreeColourFlagChallenge` method to print a 3x3 grid of nine different countries flag with their names centred below them. You must have both vertical and horizontal striped flags in your grid.

Gallery of triband flags

Program 2: Temperature Analyser

Construct a `TemperatureAnalyser` program with several methods for analysing a list of readings of temperature levels over the course of a day.

There are several things about the temperature levels that a user may be interested in:

- The average temperature level.
- How the temperatures rose and fell over the day.
- The maximum and the minimum temperature levels during the day.

The `doAnalyse()` method reads a sequence of levels from the user, using the `UI.askNumbers()` method. The sequence of levels is terminated by the word "done". It then calls 4 methods, that each do one part of the analysis.

You can run the `TemperatureAnalyser` demo program in the demonstrations folder to see what the methods should do. The demo has two buttons to help you see what needs to be done for the Core and for the Completion. Your program only needs to have one button.

Core:

- Complete the `printAverage` method that prints out the average of the temperature levels.
- Complete the `plotLevels` method so that it plots the bars in the graphics pane.

You may assume that all the temperature levels are positive. For the core, it doesn't matter if a bar goes over the top of the window.

Completion:

- Extend the `plotLevels` method so that it also draws a horizontal line for the x-axis (or baseline). You do not need any labels on the axis.
- Extend the `plotLevels` method so that it handles temperature levels over the maximum (400) by plotting a bar of the maximum size and putting an asterisk ("*") above it to show that it has been cut

off.

- Complete the `maximumOfList` method that finds and returns the maximum levels in the list.
- Complete the `minimumOfList` method that finds and returns the minimum levels in the list.

Challenge:

- Extend the `plotLevels` method to draw both an x-axis and a y-axis, and put labels on them, roughly every 50 pixels.
- Extend the `plotLevels` method to draw negative temperature levels correctly
- Extend the `plotLevels` method to use the maximum (and minimum?) temperature level to scale the y-axis so that the maximum level would reach just to the top of the graph.
- Extend the `plotLevels` method to use the number of temperature levels to scale the x-axis so that all the bars fit in the window.

Hints:

The *average* of a set of levels is the sum of the levels, divided by the number of levels.

To find the *minimum* and the *maximum*, your loop will need to keep track of the smallest and largest levels seen so far.

You may want to use the values `Double.MAX_VALUE` and `Double.MIN_VALUE` which are the largest and smallest numbers that Java recognises.

When designing and constructing a program, it is usually a good idea to get the basic part of the program working before adding the extra features. It would be a good idea to first make the `doAnalyse` method just read the levels and print out the *average*, and let the other methods print a message. Once you have that working, you can extend the program to also deal with the *plotting*, the *minimum*, and the *maximum*.

Final Part: Reflection

Think about the following questions:

1. The Core version of `drawThreeColourFlag` needed 5 parameters; the Completion version needed 6 parameters. Suggest another parameter you could add to make it even more flexible. How could you decide when to stop adding parameters to a method to make it more flexible?
2. `printAverage` computed and printed the average, but `minimumOfList` just computed and returned the minimum - it didn't print it out. Which design is better and why? What are the advantages of making a method do multiple things, and what are the advantages of making a method do just one thing?