MARKED Exercise Sheet

CE152

MARKED LAB EXERCISE 3

Introduction

This sheet contains a ***marked*** exercise. The marked exercise requires you to **submit** the project as a zip file to FASER (CE152, Lab Exercise Week 24). You must also **demonstrate** your program to one of the GLAs or module supervisors. Please check that your mark is recorded correctly. You can demonstrate your solution during Week 24 or Week 25. This marked exercise will constitute 10% of your total grade.

This exercise will focus on **Unit 7 (Lists, Abstraction), 8 (Sets, Maps) and 9 (GUI Events)**. Please watch the corresponding videos on Moodle in preparation.

You are expected to be aware of the content of these Units during your demonstration. If you are unable to explain your code this will affect your mark.

Please remember that submitting someone else's work or work generated by an AI as your own may be **considered an academic offence**. Refer to the appropriate guidelines or ask if you are uncertain about what this means.

Create today's project

Start IntelliJ and create a project named markedLab3.

MARKED EXERCISE

This sheet requires you to complete Exercise 1 [40%], Exercise 2 [50%] comment your code [5%] and export it as an archive [5%].

Exercise 1 [40%]

This exercise is more open-ended than previous exercises. The goal is to design a class that contains information about an item, a concept or an idea that exists outside of Java. For example it could describe companies, music albums, dog breeds or birthday cards. Feel free to choose something completely different. Your task is to design classes according to the requirements laid out below.

Submitting copies of code from the lecture is not permitted. Thus, avoid classes Employee, Person, Student and Card or other examples.

Exercise 1A [15%]

Create a class called named in accordance to the information it will contain that implements `Comparable`.

Your class should store at least three variables. At least one must be alphabetic, and at least one must be numeric. Use access modifiers as you deem appropriate.

Your class should have a constructor that initialises at least three variables using parameters.

The `compareTo` method should sort first by the alphabetic variable, the rest of the sequence is up to you but at least three variables must be used.

Additionally override the `toString` method to print the variables separated by a single space.

Exercise 1B [10%]

Create a class to convert user input into instances of your class. This class should have a `main()` method.

In the main method read one line of user input and split it into separate strings using one or more spaces.

Use the strings to instantiate your class as an object.

Add this object to an `ArrayList` containing objects of your class.

This should run in a loop until the program is terminated by the user.

Exercise 1C [15%]

Add the following functionality:

- Detect if more or less than three values are entered by the user and print an error if this is the case
- Detect if the user enters a non-numerical value for a numerical variable and print an error if this is the case
- Detect if the user enters `quit` and if they do print the **sorted** list of employees and exit the loop

Exercise 2 [50%]

Exercise 2A [5%]

Create a class called `MyComponent`. Extend this class from `JComponent`.

Create a class called `MyMain`. Add a `main` method to this class.

Exercise 2B [5%]

Instantiate a `JFrame` in your `main` method and add a variable of type `MyComponent` to the `JFrame`. Ensure the window is visible and will exit the program upon closing. Set the size of the window to a dimension of your choosing.

Exercise 2C [20%]

Add a mouse listener and a key listener to `MyComponent`. To verify that the events are being processed print the mouse button and the key that was pressed to the command line.

Exercise 2D [20%]

Whenever a mouse button is clicked or key on the keyboard pressed determine in which quadrant of the window the mouse cursor is. The top left is quadrant one, top right quadrant two, bottom right quadrant three and bottom left quadrant four. Store number of times an event occurred over a certain quadrants in a `Map` data structure. Whenever an event occurs print the `Map` to the command line.

For example if a mouse click or keyboard button press event occurs over quadrant one 3 times, quadrant two 2 times, quadrant 3 five times and quadrant four once:

```
{1=1}
{1=2}
{1=3}
{1=3, 2=1}
{1=3, 2=2}
{1=3, 2=2, 3=1}
{1=3, 2=2, 3=2}
{1=3, 2=2, 3=3}
{1=3, 2=2, 3=4}
{1=3, 2=2, 3=5}
{1=3, 2=2, 3=5, 4=1}
```

If the sequence is different but events occur the same amount of times over the same quadrants the end result will be the same:

```
{3=1}
{1=1, 3=1}
{1=2, 3=1}
{1=3, 3=1}
{1=3, 2=1, 3=1}
{1=3, 2=1, 3=1, 4=1}
{1=3, 2=2, 3=1, 4=1}
{1=3, 2=2, 3=2, 4=1}
{1=3, 2=2, 3=3, 4=1}
{1=3, 2=2, 3=4, 4=1}
{1=3, 2=2, 3=5, 4=1}
```

See Figure 1 for an example of the quadrants. You window does not need to display the quadrants.

Comments [5%]

In this exercise you should practise commenting your code. To receive the marks please provide **one comment per method** explaining the **purpose** of the method and your **approach** to providing the requested functionality.

Exporting and submitting [5%]

Please read this entire section before submitting your code to Faser.

Using IntelliJ, in the menu File, Export choose Project to zip file. . . . Note the folder you exported to and submit this zip file to Faser. **Please check that you have uploaded the correct file**: download it from Faser, unzip it and check if the source code you wrote is contained in the archive.

Some versions of IntelliJ do not have the export to zip file item on the menu. In this case, you have two options:

1. File, Settings. . . , go to the Plugins tab, Installed pane, and tick the Android plugin. When you hit OK, IntelliJ will restart and the menu item should be there.
2. Just use any zip software to create a zip file of the project folder. Include the whole project – this is one level above `src` in a project that's set up in the standard way.

If you are not using IntelliJ you must still upload your project to Faser as a zip archive. Unfortunately, I cannot provide you with instructions on how to do this, but you can always use any zip software, as above.

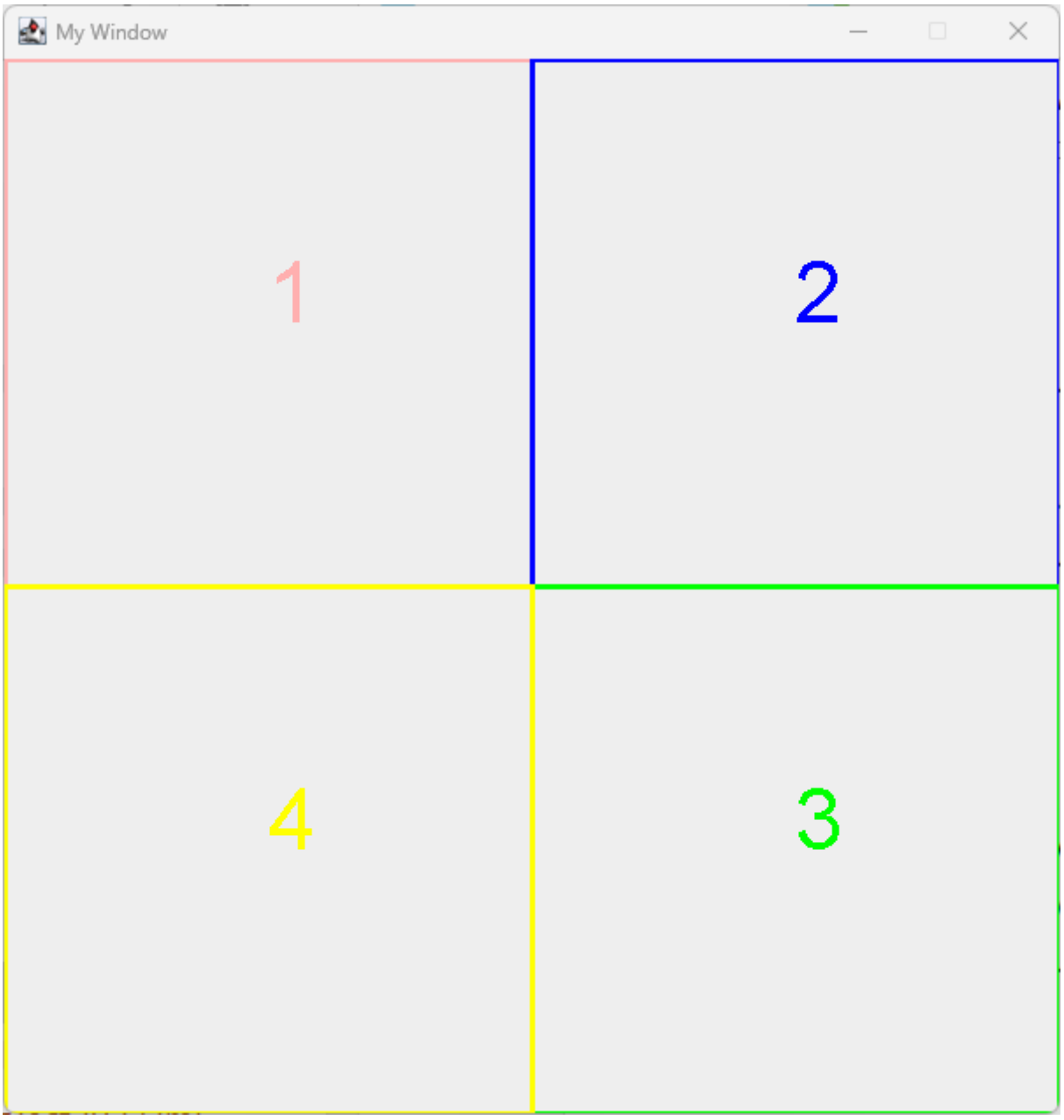

Figure 1: An example JFrame illustrating the numbering of the quadrants. You JFrame does not need to display them.