CSC8001 (Programming and Data Structures)
Practical session 2: Creating projects.

In this practical session you will create, design and implement java code in three different projects.

The learning outcomes for the session are:

- learn how to create your own Java project using **BlueJ**
- design and code your own programs
- gain experience with control flow (if statements)

If you have any questions during the following exercises, please ask one of the demonstrators for help!

0. Preliminaries

This session assumes you have completed the familiarisation exercises contained in the first practical session. In particular, you should be familiar with running BlueJ on a Windows PC, opening projects, creating objects, using the Object Bench to execute methods and inspect objects, and editing class definitions. If you need a reminder, consult the previous practical session information.

1. New Project: Degree classifications

Your first task will be to write a class to calculate degree classifications, using an **if** statement.

First, start BlueJ and begin a new project, by selecting **New Project** from the **Project** menu. This will give you a file selection dialog to navigate to and select the folder in which your new project will be created. It's a good idea to create your new project within your CSC8001 folder. When you have selected this folder, type the name of your new project in the box provided, and click on **Create**. In this case, a name such as **Degrees** would be a good choice.

You are now presented with a new project window, in which there is only a document icon (README file). Opening this up, you will see some initial documentation. This is your opportunity to create the documentation for the project. You can return to it later to update the information.

Now create a new class. Click on the **New Class...** button. This will give you another dialog box, where you supply the name of the class you are about to create, and select the type of class you want. The name of the class should be **Degree**, and type of class the default case: create a Class.

Open the new **Degree** class which appears, and edit the template code that you find there to create the methods for your new class. Make sure you document the code with comments.

In this case, there are no instance variables and the constructor will have nothing to do.

Your class should have a single method, **Classify**, which takes as input a single integer mark between 0 and 100 and returns a **String**, either "Fail", "Pass", "Merit", "Distinction" or "Out of range". The value of the string depends on the value of the mark:

- A mark less than 0 or greater than 100 is "Out of range".
- A mark of zero or more, but less than 50 is "Fail".
- A mark of 50 or more, but less than 60 is "Pass".
- A mark of 60 or more, but less than 70 is "Merit".
- A mark of 70 or more, but less than 100 is "Distinction".

Think about what values you should use to test your **Classify** methods and make sure it behaves as you expect. Ensure you have documented your project by updating the README file and commenting the code.

2. The Biggest and the Smallest

In a new project, create a class called **MaxAndMin** whose constructor takes a single integer parameter, which becomes the initial value of a instance variable called **value**.

Provide two methods:

- **fetch** which has no parameters, but which returns the current value of **value**, and
- **store**, which has a single parameter and returns no value. This method updates the field **value** by setting it to the value of the method's parameter.

Now add some facilities to the class which allow an object of the class to remember some of its past "history", by saving the largest and smallest values which have ever been stored in it. Add two methods to the class which return:

- the maximum value that the field **value** has taken since the creation of the object, and
- the minimum value that the field **value** has taken since the creation of the object.

Clearly, immediately after an object of this class has been created, both of these methods will return the value supplied as the parameter to the constructor.

Think about what values you should use with **store** to test that your methods for recording the maximum and minimum are working correctly.

Ensure you have documented your project by updating the README file and commenting the class and its methods.

For any questions, ask a demonstrator during the practical sessions, or email:
steve.riddle@ncl.ac.uk