Queen's University Belfast

CSC1025

LEVEL 1

Procedural Programming

MONDAY, 20th October 2025

Examiners: Dr Darryl Stewart

and the internal examiners

Statement of Integrity

By submitting the work, I declare that:

1. I have read and understood the University regulations relating to academic offences, including collusion and plagiarism: http://www.qub.ac.uk/directorates/AcademicStudentAffairs/AcademicAffairs/GeneralRegulations/Procedures/ProceduresforDealingwithAcademicOffences/
2. The submission is my own original work and no part of it has been submitted for any other assignments, except as otherwise permitted;
3. All sources used, published or unpublished, have been acknowledged;
4. I give my consent for the work to be scanned using a plagiarism detection software.

Module Learning Outcomes Addressed

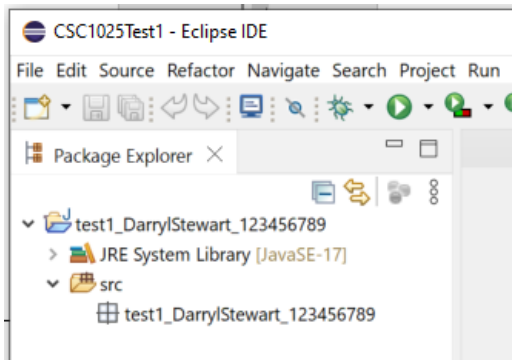
1. Demonstrate knowledge, understanding and the application of the principles of procedural programming, including:
    - Primitive data types (including storage requirements) **[Partial]**
    - Program control structures: Sequencing, selection and iteration **[Partial]**
    - Functions/methods and data scope **[Not Addressed]**
    - Simple abstract data structures, i.e. Strings **[Partial]**
    - File I/O and error handling **[Not Addressed]**
    - Pseudocode and algorithm definition/refinement **[Not Addressed]**
2. Apply good programming standards in compliance with the relevant codes of practice e.g. naming conventions, comments and indentation **[Partial]**
3. Analyse real-world challenges in combination with programming concepts to write code in an effective way to solve the problem. **[Partial]**

Instructions

- Test 1 is worth 30% of the module marks
- Test 1 - Part B is worth 50% of Test 1
- Adding comments to your code is encouraged to explain your solution and they may be helpful during the marking process if the code doesn't work correctly
- You **MUST** use appropriate variable names and use correct code indentation.
- You may use the Canvas course notes during this part of the test but no videos or practical solutions.
- You may also refer to up to two sides of pre prepared A4 notes.
- You may also use pen and paper or a calculator if this is useful.

Initial setup

1. Open Eclipse and select "Switch Workspace"
   - Create a new folder on either your ***OneDrive*** or on the ***I:Drive*** called **CSC1025Test1** and select this as your workspace
2. Create a Java project named test1_<your name_your student id>, e.g.
   test1_DarrylStewart_123456789
3. Create a java package in your ***src*** folder with the **same name** as the project. E.g. it should look like this:



4. Add your new Class to this package to complete the **Animal Shelter** programming task detailed on the following pages.

Part B – Programming Task

Animal Shelter Weight Analyzer

Write a single Java program called **Animals** that prompts the user for the number of animals to track, validates that input, then processes the weights of each animal, calculating totals, max weights, and averages.

Part 1: Input Validation [30 marks]

- Ask the user to enter the number of animals to track. This will be a whole number.
- Use an appropriate loop to repeatedly prompt the user until a valid number is provided.
    - Input is valid if it is between 2 and 10 inclusive.
    - If input is invalid, output a specific error message: "Error: Number of animals must be between 2 and 10."

Part 2: Data Entry and Calculation [60 marks]

- Once a valid number of animals is confirmed in Part 1, the program must collect the weight for each animal and perform the required analysis.
- Use a for loop for the following stages of the program to iterate for each animal (based on the number of animals entered in Part1). It should prompt the user to enter the weight of each animal (in kg), e.g. 5.6kg. Using these values it should:
    - Calculate the total sum of the weights
    - Store the weight of the lightest animal
    - Calculate the average weight, where:

      Average Weight = Total sum of the weights/ Number of animals.
    - Count the total number of animals that are underweight or overweight. Underweight animals are less than 5.0kg and overweight animals are over 15.0kg.

Part 3: Output Results [10 marks]

- Display the total sum of the weights formatted to two decimal places and including "kg" using printf.
- Display the average weight formatted to two decimal places and including "kg" using printf.
- Display the lightest weight recorded to one decimal place.
- Display the count of how many animals are under/overweight.

The output should be **formatted** and **presented precisely** as shown in the following sample output

Expected Output Example :

User inputs: **1, 15,** then **5;** Weights provided are **18.0,4.0,10.0,20.0,4.5**. The green text indicates the text entered by the user

```
Enter the Number of Animals to Track (2-10): 1
Error: Number of animals must be between 2 and 10.
Enter the Number of Animals to Track (2-10): 15
Error: Number of animals must be between 2 and 10.
Enter the Number of Animals to Track (2-10): 5

Enter Weight of Animal 1 (in kg): 18.0
Enter Weight of Animal 2 (in kg): 4.0
Enter Weight of Animal 3 (in kg): 10.0
Enter Weight of Animal 4 (in kg): 20.0
Enter Weight of Animal 5 (in kg): 4.5

--- Analysis Complete ---
Total Weight Sum: 56.50 kg
Average Weight: 11.30 kg
Lightest Weight Recorded: 4.0 kg
Number of Animals under/overweight: 4
```