BSc (Honours) in Computing in Software Development
BSc (Honours) in Computing in Games Development
Stage 2

Module:	Object Oriented Programming	**Assignment:**	CA1
Due:	see Moodle	**Credit:**	10%

Junior Certificate (JC) Results Weighting Problem

Objective

You are required to implement and test a Java program to generate a weighting value, based on Junior Certificate results, to be used in the calculation of leaving certificate grades.

Specification

The program is required to calculate a straight average of five marks from the following subjects:

- Mathematics, English and Irish
  AND
- The **two highest marks** from the remaining subjects with the exception of CSPE (CSPE is not to be included)

The program should output a list of student numbers and their corresponding calculated average value. e.g. 891234 56.54

Student Data File

Student data will be read from a file called “JC_Results.txt” with the structure shown below.

Student_Number, Subject_Code, Mark, Subject_Code, Mark, … (newLine)

Each line represents one student record.
Each student will have exactly **eight** “Code,Mark” pairs. (To reduce complexity)
Subject Codes may appear in **any order** for a student record.
You may assume that the structure of the data in the file is correct (so no need to validate)

```
students_jc_results.txt  (sample data)
891234,1,65,2,58,3,45,4,60,5,50,12,48,42,42,46,60
783461,3,65,1,58,2,45,125,60,137,68,126,100,57,77,4,60
Etc…
```

Subject codes are available in the Excel file “JC Codes.xls”. This information does not need to be read into your program, but the correct codes must be used in the logic of your program.

The following two methods **must** be used in the program:

```
    public int[] selectFiveGrades(int[] codes, int[] grades) {  …
```

// to accept the eight available codes and corresponding grades, and return an array of the five selected grades (based on the selection algorithm)

```
    private double calculateAverage( int[] selectedGrades) { …
```

// to calculate the average

(The above methods will lend themselves to unit testing)

Testing

Devise an appropriate set of tests for this program, and implement those tests using **Junit**.

Grading Criteria

Functionality	60%
Code Quality	20%
Testing	10%
Source Code Repository (Git)	10%

Submission Requirements

1. Students must type their name and student ID at the top of **every** source code file.
2. Source code, and other items must be submitted in the relevant sub-folders inside a single ZIP file through Moodle.
3. Students may be required to attend an **interview** after the deadline date.
4. The assignment must be entirely the work of each student. Student are not permitted to share any pseudocode or source code from their solution with any other individual in the class. Students may not distribute the source code of their solution to any student in any format (i.e. electronic, verbal, or hardcopy transmission).
5. **Plagiarised assignments will receive a mark of zero**. This also applies to the individual allowing their work to be plagiarised.
6. Any plagiarism will be reported to the Head of Department and a report will be added to your permanent academic record.
7. Late assignments will only be accepted if accompanied by the appropriate medical note. This documentation must be received within 10 working days of the project deadline. The penalty for late submission is as follows:
    - Marked out of 80% if up to 24 hours late.
    - Marked out of 60% if 24-48 hours late.
    - Marked out of 40% if 48-72 hours late.
    - Marked out of 20% if 72-96 hours late.
    - Marked out of 0%, if over 96 hours late.