# Introduction to Programming COIT 20245

## Assessment item 1—Java Console Program

| | | |
|---|---|---|
| **Due date:** | Week 5 Friday (12 Aug 2022) 11:59 pm AEST | **ASSESSMENT** |
| | Refer below for complete assessment item 1 requirements (**Assignment One**) | |
| **Weighting:** | 20% | |
| **Length:** | N/A | **1** |

## Objectives

This assessment item relates to the course learning outcomes as in the Unit Profile.

## Details

For this assignment, you are required to develop a **Java Console Program** to demonstrate you can use Java constructs including input/output via a command line, Java primitive and built-in data types, Java defined objects, selection and looping statements, methods, and various other Java commands. Your program must produce the correct results.
**You are only allowed to use techniques which have been covered in the first five weeks of the course**, you must use the **Scanner** object for input and **no advanced data structures like arrays** will be used. Instructions for this appear in the implementation section of this specification.

**What to submit for this assignment?**

The Java source code:

- **GradingSystem.java**

A report including a flow chart (UML activity diagram) to depict your validation loop for reading the number of students, how long it took to create the program, any problems encountered, and screenshots of the output produced. (Use Alt-PrtScrn to capture just the console window and you can paste it into your Word document). You should thoroughly test your program.
**Important**: For this assignment, you are required to paste your source code as an appendix into your report, do not worry about the formatting loss.

- **ReportAss1.docx**

You will submit your files by the due date using the "**Assignment 1 Submission**" link on the Moodle unit website in the **Assessment Block** or in the relevant week.

# Assignment specification

**COIT20245 Grading System**

In COIT20245, you have three assessments with maximum weighting as below.

```
ASSIGNMENT_ONE = 20;
ASSIGNMENT_TWO = 30;
FINAL_PROJECT = 50;
```

You are to write a **Java Console Application** (GradingSystem.java) which will allow students to enter the details of N students names, student numbers and expected marks of all three assessments of COIT20245. **N should be equal to the highest digit in your student ID**, use N=3 if your highest digit is less than three. For each record the program will prompt for and accept the student's name, student number and expected marks of all three assessments of COIT20245, it will then display the grade (F, P, C, D, HD) as per grading requirements below.

F: totalMarks <50
P: (totalMarks>=50) or (totalMarks <65)
C: (totalMarks>=65) or (totalMarks <75)
D: (totalMarks>=75) or (totalMarks <85)
HD: (totalMarks>=85)

When N students' marks have been entered you need to report the maximum and minimum number of total marks (sum of all assessment's mark) and the relevant students name and student number, the average number of total marks for the entire class.

The required **Java Console Application** should allow the user to:

1. For each of the N records: enter the ***student's name***, student's number and then enter marks of all assessments. The program will output the grade of the student.
2. You must ensure the student's name and student's number is not blank so you must implement a validation loop to ensure that a student's name and student's number are entered. For this assignment, there is no need to ensure the name or number is a valid entry (e.g. entering 1 for the name would be allowed). The marks must be greater than zero and less than maximum weighting allocated for the assignment.

The program will number each entry in the input prompt.

3. When N records have been entered, you will output a heading for the statistics "Statistical information for COIT20245 grading system", the minimum and maximum number of marks and the students names with these minimums and maximums, and then what the average marks (see sample output below). Note: If more than one student has an equal maximum or minimum marks you just need to only output one students detail.
4. Display a welcome message at the beginning "Welcome to COIT20245 grading system" and an end message e.g. "Thank you for using the COIT20245 grading system" and the final line "Program written by <your student ID>" (see sample output below).

The numeric literal values N, number of students, grades and assessment marks ***must*** be represented as **constants**.

# Implementation

A large number of students have never written a program before so this is a fairly simple assignment which can be written in the main method of your class. Follow the steps outlined here and build your program up in a step by step fashion and always compile your program at each stage so you are always working on error-free code.

Start by creating your GradingSystem class which will contain just the main method, COMPILE! (Fix any errors and repeat)

Implement the welcome message, COMPILE, RUN and TEST!

Declare your Scanner object(s), COMPILE!

> Note: In order to combat the problem of the Scanner objects reading both textual and numeric data a good way to counter this is to declare two Scanner objects, one for reading text and another for reading numbers, or you can clear the buffer after the int read using `nextLine()`

Create a loop to loop N Times, COMPILE, RUN and TEST! (use N = 3 for development)

Declare variables to hold the student's name, number and assessment marks (String,String,int), COMPILE, RUN and TEST!

Within the loop: prompt and read the student's name and student number, COMPILE, RUN and TEST!

Add the prompt and read the marks of the assessments, COMPILE, RUN and TEST!

Calculate the grade using conditions above, COMPILE, RUN and TEST until this is correct.

Output the description of the student's grade (see sample output below)

Add the validation loops for reading the data (you can do this last if you like)

Use if statements to determine if the number of total marks is maximum or minimum, (you will have to think about this). Output the minimum and maximums after the loop, COMPILE and RUN until you have this correct. You may want to set your original max and min variables to very small and very large numbers using `Integer.MIN_VALUE` and `Integer.MAX_VALUE.`

You will need to add up the total number as you go so you can calculate the average.
After the loop, you will output the statistics which you have gathered in the loop. You should have the maximum and minimum number ls and the corresponding student's names, number and the total marks.
Output the statistics as indicated in the screen shot below.
Finally, print the end message.

Your program should be well laid out, commented and use appropriate and consistent names (camel notation) for all variables and objects. Meaningful comments need to be placed before classes, methods and in the body of the code including variables.
**For this assignment, you will not worry about checking data types**.
Refer to a Java reference textbook and the unit and lecture material (available on the course WEB site) for further information about the Java programming topics required to complete this assignment.
Check the marking guide (last page) to ensure you have completed every task. You need to match the output as shown below including line spacing.

Typical screen shot of the program executing is as follows:


----------Good luck----------

# Marking Scheme

| | Marks allocated |
|---|---|
| **Total number of marks – 20** | |
| **Variables, constants and types** | |
| Constant N is used correctly and is largest digit of student ID | 0.5 |
| Constants are used for all numeric literals (no hard-coded values) | 0.5 |
| Variables have meaningful names and use camel notation | 0.5 |
| Variables are the correct type | 0.5 |
| **Code in general** | |
| Code is indented and aligned correctly | 0.5 |
| Code has header comment which includes student name, student ID, date, file name and purpose of the class | 0.5 |
| Code is fully commented including all variables | 0.5 |
| **Input** | |
| Numbers and strings are read correctly | 0.5 |
| Validation loop for student's name and student number is correct | 2 |
| Validation loop for marks is correct | 1 |
| **Processing** | |
| if statements are correct | 0.5 |
| looping is correct i.e iterates N times | 0.5 |
| **Output** | |
| Output is formatted correctly (resembles sample output) | 1.75 |
| Correct grade for each student is correct | 1.75 |
| Minimum totalMark and relevant student name with student number are correct | 1 |
| Maximum totalMark and relevant student name with student number are correct | 1 |
| Average totalMark is correct | 1 |
| Total marks are correct for each student | 0.5 |
| Welcome and exit message (with student ID) are displayed | 0.5 |
| **General** | |
| Correct files submitted including types and names | 1 |
| Only techniques covered during weeks 1-6 are used | 1 |
| **Report** | |
| Flow chart is correct | 1 |
| Screen shot(s) of testing and annotations | 0.5 |
| Report presentation and comments including how long it took and any problems encountered | 0.5 |
| Source code has been included as an appendix | 0.5 |