



Te Pūkenga

College of Engineering, Construction & Living Sciences

Bachelor of Information Technology

ID511001: Programming 2

Level 5, Credits 15

Project 1 (C# Console App): Learner Gradebook

Assessment Overview

In this assessment, you will design & develop a learner gradebook **Console App** using **C#**.

Learning Outcomes

At the successful completion of this course, learners will be able to:

1. Build interactive, event-driven GUI applications using pre-built components.
2. Declare & implement user-defined classes using encapsulation, inheritance & polymorphism.

Assessments

Assessment	Weighting	Due Date	Learning Outcomes
Project 1 (C# Console App): Learner Gradebook	25%	26-04-2023 (Wednesday at 4.59 PM)	1 & 2
Project 2 (C# Windows Forms App): Pong	35%	14-06-2023 (Wednesday at 4.59 PM)	1 & 2
Theory Examination	30%	21-06-2023 (Wednesday at 4.45 PM)	1 & 2
Classroom Tasks	10%	07-06-2023 (Wednesday at 4.59 PM)	1 & 2

Conditions of Assessment

You will complete this assessment during your learner-managed time. However, there will be time during class to discuss the requirements & your progress on this assessment. This assessment will need to be completed by **Wednesday, 26 April 2022 at 4.59 PM**.

Pass Criteria

This assessment is criterion-referenced (CRA) with a cumulative pass mark of **50%** over all assessments in **ID511001: Programming 2**.

Authenticity

All parts of your submitted assessment **must** be completely your work. If you use code snippets from **GitHub**, **StackOverflow** or other online resources, you **must** reference it appropriately using **APA 7th edition**. Provide your references in the **README.md** file in your repository. Failure to do this will result in a mark of **zero** for this assessment.

Policy on Submissions, Extensions, Resubmissions & Resits

The school's process concerning submissions, extensions, resubmissions & resits complies with **Te Pūkenga** policies. Learners can view policies on the **Te Pūkenga** website located at <https://www.op.ac.nz/about-us/governance-and-management/policies>.

Submission

You **must** submit all app files via **GitHub Classroom**. Here is the URL to the repository you will use for your submission – <https://classroom.github.com/a/eFe1Oh97>. Create a **.gitignore** & add the ignored files in this resource – <https://raw.githubusercontent.com/github/gitignore/main/VisualStudio.gitignore>. The latest app files in the **master** or **main** branch will be used to mark against the **Functionality** criterion. Please test before you submit. Partial marks **will not** be given for incomplete functionality. Late submissions will incur a **10% penalty per day**, rolling over at **5:00 PM**.

Extensions

Familiarise yourself with the assessment due date. Contact the course lecturer before the due date if you need an extension. If you require more than a week's extension, you will need to provide a medical certificate or support letter from your manager.

Resubmissions

Learners may be requested to resubmit an assessment following a rework of part/s of the original assessment. Resubmissions are to be completed within a negotiable short time frame & usually **must** be completed within the timing of the course to which the assessment relates. Resubmissions will be available to learners who have made a genuine attempt at the first assessment opportunity & achieved a **D grade (40-49%)**. The maximum grade awarded for resubmission will be **C-**.

Resits

Resits & reassessments **are not** applicable in **ID511001: Programming 2**.

Instructions

You will need to submit an app & documentation that meet the following requirements:

Functionality - Learning Outcomes 1 & 2 (40%)

- The app **must** open without code or file structure modification in **Visual Studio**.
- Read a text file called **learners.txt** which contains information about five learners. This information includes **id**, **first name**, **last name**, three **ID510001: Programming 1 assessment marks** & three **ID511001: Programming 2 assessment marks**. **Note: learners.txt must** be located in the **bin/Debug** folder.
- The learners' information is stored in a **List** of **Learner** objects. A **Learner** object **must** have the following fields:
 - **id**
 - **firstName**
 - **lastName**
 - **AssessmentMarks** object called **prog1AssessmentMarks**
 - **AssessmentMarks** object called **prog2AssessmentMarks**

Note: A **Learner** has no behaviours.

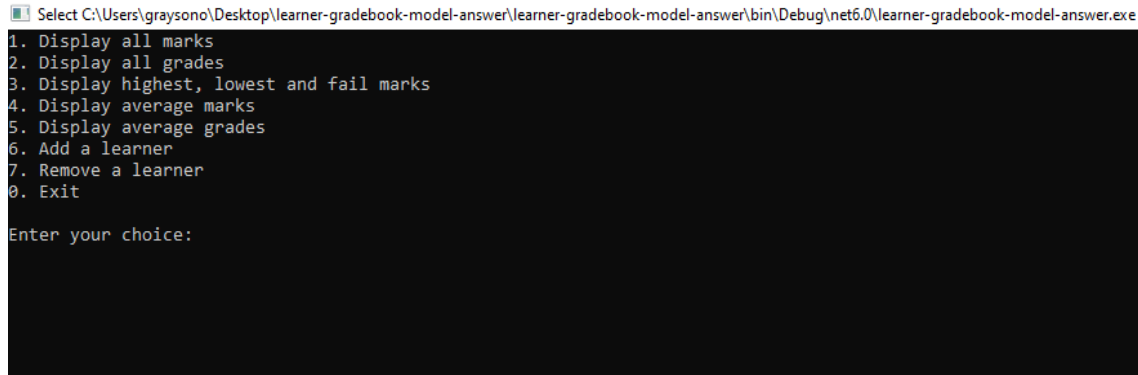
- An **AssessmentMarks** object **must** have the following fields:
 - A **List** of **int** called **assessmentMarks**
- An **AssessmentMarks** object has several behaviours such as getting all marks, all grades, highest mark(s), lowest mark(s), fail mark(s), average marks and average grades.
- A grade is calculated using the following grade table:

Grade	Mark Range
A+	90-100
A	85-89
A-	80-84
B+	75-79
B	70-74
B-	65-69
C+	60-64
C	55-59
C-	50-54
D	40-49
E	0-39

- The app **must** display the following menu options:
 - 1. Display all marks
 - 2. Display all grades

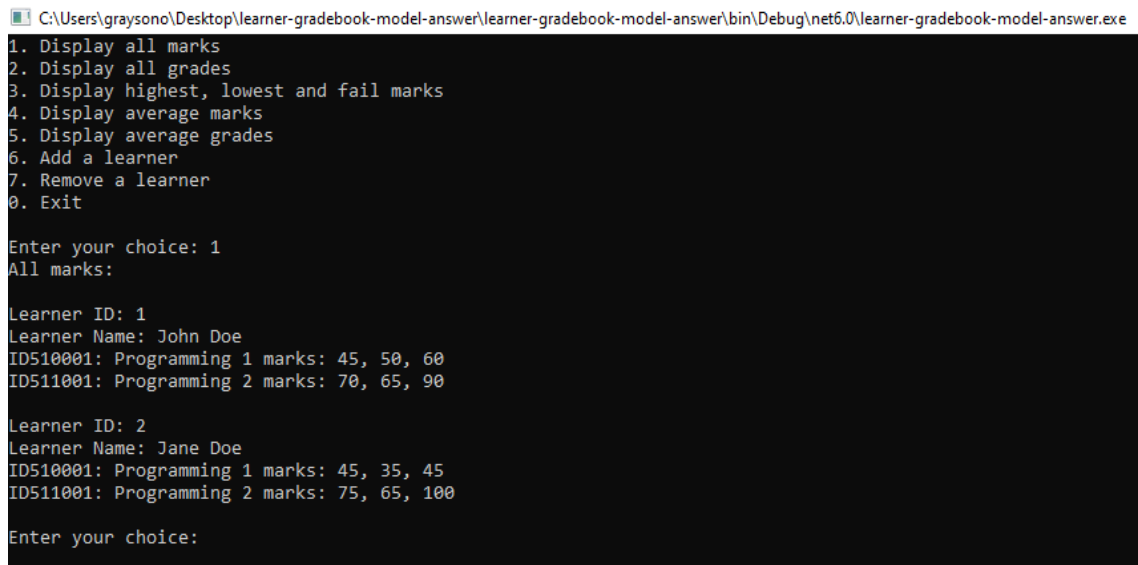
- 3. Display highest, lowest and fail marks
- 4. Display average marks
- 5. Display average grades
- 6. Add a learner
- 7. Remove a learner
- 0. Exit

Note: You will need to create individual methods to achieve this functionality. Also, the app **must** be able to handle invalid user input. If the user enters an invalid option, an error message **must** be displayed.



```
Select C:\Users\grayson\Desktop\learner-gradebook-model-answer\learner-gradebook-model-answer\bin\Debug\net6.0\learner-gradebook-model-answer.exe
1. Display all marks
2. Display all grades
3. Display highest, lowest and fail marks
4. Display average marks
5. Display average grades
6. Add a learner
7. Remove a learner
0. Exit
Enter your choice:
```

- When the user selects **1. Display all marks**, the app **must** display all marks for all learners. The marks **must** be displayed as follows:



```
C:\Users\grayson\Desktop\learner-gradebook-model-answer\learner-gradebook-model-answer\bin\Debug\net6.0\learner-gradebook-model-answer.exe
1. Display all marks
2. Display all grades
3. Display highest, lowest and fail marks
4. Display average marks
5. Display average grades
6. Add a learner
7. Remove a learner
0. Exit
Enter your choice: 1
All marks:
Learner ID: 1
Learner Name: John Doe
ID510001: Programming 1 marks: 45, 50, 60
ID511001: Programming 2 marks: 70, 65, 90
Learner ID: 2
Learner Name: Jane Doe
ID510001: Programming 1 marks: 45, 35, 45
ID511001: Programming 2 marks: 75, 65, 100
Enter your choice:
```

- When the user selects **2. Display all grades**, the app **must** display all grades for all learners. The grades **must** be displayed as follows:

```
C:\Users\grayson\Desktop\learner-gradebook-model-answer\learner-gradebook-model-answer\bin\Debug\net6.0\learner-gradebook-model-answer.exe
1. Display all marks
2. Display all grades
3. Display highest, lowest and fail marks
4. Display average marks
5. Display average grades
6. Add a learner
7. Remove a learner
8. Exit

Enter your choice: 2

All grades:

Learner ID: 1
Learner Name: John Doe
ID510001: Programming 1 grades: D, C-, C+
ID511001: Programming 2 grades: B, B-, A+

Learner ID: 2
Learner Name: Jane Doe
ID510001: Programming 1 grades: D, E, D
ID511001: Programming 2 grades: B+, B-, A+

Enter your choice:
```

- When the user selects **3. Display highest, lowest and fail marks**, the app **must** display the highest, lowest and fail marks for all learners. The marks **must** be displayed as follows:

```
C:\Users\grayson\Desktop\learner-gradebook-model-answer\learner-gradebook-model-answer\bin\Debug\net6.0\learner-gradebook-model-answer.exe
1. Display all marks
2. Display all grades
3. Display highest, lowest and fail marks
4. Display average marks
5. Display average grades
6. Add a learner
7. Remove a learner
8. Exit

Enter your choice: 3

Highest, lowest and fail marks:

Learner ID: 1
Learner Name: John Doe
ID510001: Programming 1 highest marks: 60
ID510001: Programming 1 lowest marks: No lowest marks
ID510001: Programming 1 fail marks: 45
ID511001: Programming 2 highest marks: 90
ID511001: Programming 2 lowest marks: No lowest marks
ID511001: Programming 2 fail marks: No fail marks

Learner ID: 2
Learner Name: Jane Doe
ID510001: Programming 1 highest marks: 45, 45
ID510001: Programming 1 lowest marks: No lowest marks
ID510001: Programming 1 fail marks: 45, 35, 45
ID511001: Programming 2 highest marks: 100
ID511001: Programming 2 lowest marks: No lowest marks
ID511001: Programming 2 fail marks: No fail marks

Enter your choice:
```

Note: If there is no fail mark(s), the **Fail marks: must** be displayed as **No fail marks**. If there is no lowest mark(s), the **Lowest marks: must** be displayed as **No lowest marks**. If there is no highest mark(s), the **Highest marks: must** be displayed as **No highest marks**.

- When the user selects **4. Display average marks**, the app **must** display the average marks for all learners. The marks **must** be displayed as follows:

```
C:\Users\grayson\Desktop\learner-gradebook-model-answer\learner-gradebook-model-answer\bin\Debug\net6.0\learner-gradebook-model-answer.exe
1. Display all marks
2. Display all grades
3. Display highest, lowest and fail marks
4. Display average marks
5. Display average grades
6. Add a learner
7. Remove a learner
0. Exit

Enter your choice: 4

Average marks:

Learner ID: 1
Learner Name: John Doe
ID510001: Programming 1 average mark: 51.67
ID511001: Programming 2 average mark: 75

Learner ID: 2
Learner Name: Jane Doe
ID510001: Programming 1 average mark: 41.67
ID511001: Programming 2 average mark: 80

Enter your choice:
```

- When the user selects **5. Display average grades**, the app **must** display the average grades for all learners. The grades **must** be displayed as follows:

```
C:\Users\grayson\Desktop\learner-gradebook-model-answer\learner-gradebook-model-answer\bin\Debug\net6.0\learner-gradebook-model-answer.exe
1. Display all marks
2. Display all grades
3. Display highest, lowest and fail marks
4. Display average marks
5. Display average grades
6. Add a learner
7. Remove a learner
0. Exit

Enter your choice: 5

Average grades:

Learner ID: 1
Learner Name: John Doe
ID510001: Programming 1 average grade: C-
ID511001: Programming 2 average grade: B+

Learner ID: 2
Learner Name: Jane Doe
ID510001: Programming 1 average grade: D
ID511001: Programming 2 average grade: A-

Enter your choice:
```

- When the user selects **6. Add a learner**, the app **must** prompt the user to enter the following information:
 - **First name**
 - **Last name**
 - **ID510001: Programming 1 assessment mark 1**
 - **ID510001: Programming 1 assessment mark 2**
 - **ID510001: Programming 1 assessment mark 3**
 - **ID511001: Programming 2 assessment mark 1**
 - **ID511001: Programming 2 assessment mark 2**
 - **ID511001: Programming 2 assessment mark 3**

Note: An **assessment mark** must be between **0** and **100**. If an **assessment mark** is invalid, an error message **must** be displayed. Also, the **id** of the learner **must** be generated automatically. However, the **id** **must** be unique.

```
C:\Users\grayson\Desktop\learner-gradebook-model-answer\learner-gradebook-model-answer\bin\Debug\net6.0\learner-gradebook-model-answer.exe
1. Display all marks
2. Display all grades
3. Display highest, lowest and fail marks
4. Display average marks
5. Display average grades
6. Add a learner
7. Remove a learner
0. Exit

Enter your choice: 6

Add learner:
Enter learner's first name: Grayson
Enter learner's last name: Orr
Enter ID510001: Programming 1 assessment 1 marks: 50
Enter ID510001: Programming 1 assessment 2 marks: 60
Enter ID510001: Programming 1 assessment 3 marks: 75
Enter ID511001: Programming 2 assessment 1 marks: 80
Enter ID511001: Programming 2 assessment 2 marks: 55
Enter ID511001: Programming 2 assessment 3 marks: 10
Learner with the id: (3) was added

Enter your choice: 1
All marks:

Learner ID: 1
Learner Name: John Doe
ID510001: Programming 1 marks: 45, 50, 60
ID511001: Programming 2 marks: 70, 65, 90

Learner ID: 2
Learner Name: Jane Doe
ID510001: Programming 1 marks: 45, 35, 45
ID511001: Programming 2 marks: 75, 65, 100

Learner ID: 3
Learner Name: Grayson Orr
ID510001: Programming 1 marks: 50, 60, 75
ID511001: Programming 2 marks: 80, 55, 10

Enter your choice:
```

- When the user selects **7. Remove a learner**, the app **must** prompt the user to enter the **id** of the learner to be removed. If the learner is found, the learner **must** be removed from the **List of Learner** objects. If the learner is not found, an error message **must** be displayed.

```
C:\Users\grayson\Desktop\learner-gradebook-model-answer\learner-gradebook-model-answer\bin\Debug\net6.0\learner-gradebook-model-answer.exe
1. Display all marks
2. Display all grades
3. Display highest, lowest and fail marks
4. Display average marks
5. Display average grades
6. Add a learner
7. Remove a learner
0. Exit

Enter your choice: 7

Remove learner:
Enter learner id: 1
Are you sure you want to delete this learner? (y/n): y
Learner with the id: (1) was deleted

Enter your choice: 1
All marks:

Learner ID: 2
Learner Name: Jane Doe
ID510001: Programming 1 marks: 45, 35, 45
ID511001: Programming 2 marks: 75, 65, 100

Learner ID: 3
Learner Name: Grayson Orr
ID510001: Programming 1 marks: 50, 60, 75
ID511001: Programming 2 marks: 80, 55, 10

Enter your choice:
```

- When the user selects **0. Exit**, the app **must** exit.
- 20 **unit tests** using **MSTest** which verify the functionality.

Code Elegance - Learning Outcomes 1 & 2 (45%)

- Adhere to the two principles of **OO**, i.e., encapsulation & abstraction.
- Use of intermediate variables, constants & try-catch blocks.
- Idiomatic use of control flow, data structures & in-built functions.
- Efficient algorithmic approach.
- Sufficient modularity.
- Each method & class **must** have a header comment located immediately before its declaration.
- In-line comments where required.
- App files, i.e., **.cs** files are formatted.
- No dead or unused code.

Documentation & Git Usage - Learning Outcomes 1 & 2 (15%)

- Provide the following in your repository **README.md** file:
 - The app's class diagram created in **Visual Studio**.
 - How to run the **unit tests**.
 - Known bugs if applicable.
- Commit at least **20** times per week.
- Commit messages **must** be formatted using the recommended conventions & reflect the context of each functional requirement change.