# **ID511001: Programming 2**

# **Project 1 (C# Console App): Learner Gradebook Marking Rubric**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **10-9** | **8-7** | **6-5** | **4-0** |
| **Functionality** | The app contains comprehensive and robust evidence on the following functionality: No code/file structure modification, reading a text file of learners, getting, and displaying all marks, all grades, highest mark(s), lowest mark(s), fail mark(s), average marks, and average grades, error handling and unit testing. | The app contains clear and detailed evidence on the following functionality: No code/file structure modification, reading a text file of learners, getting, and displaying all marks, all grades, highest mark(s), lowest mark(s), fail mark(s), average marks, and average grades, error handling and unit testing. | The app contains evidence on the following functionality: No code/file structure modification, reading a text file of learners, getting, and displaying all marks, all grades, highest mark(s), lowest mark(s), fail mark(s), average marks, and average grades, error handling and unit testing. | The app does not or does not fully contain evidence on the following functionality: No code/file structure modification, reading a text file of learners, getting, and displaying all marks, all grades, highest mark(s), lowest mark(s), fail mark(s), average marks, and average grades, error handling and unit testing. |
| **Code Elegance** | The app demonstrates comprehensive evidence on the following:   * Use of OO principles, i.e., encapsulation and abstraction. * Use of intermediate variables, constants, and try-catch blocks. * Idiomatic use of control flow, data structures and in-built functions. * Efficient algorithmic approach. * Sufficient modularity. * Commenting and formatting. * No dead or unused code. | The app demonstrates clear evidence on the following:   * Use of OO principles, i.e., encapsulation and abstraction. * Use of intermediate variables, constants, and try-catch blocks. * Idiomatic use of control flow, data structures and in-built functions. * Efficient algorithmic approach. * Sufficient modularity. * Commenting and formatting. * No dead or unused code. | The app demonstrates evidence on the following:   * Use of OO principles, i.e., encapsulation and abstraction. * Use of intermediate variables, constants, and try-catch blocks. * Idiomatic use of control flow, data structures and in-built functions. * Efficient algorithmic approach. * Sufficient modularity. * Commenting and formatting. * No dead or unused code. | The app does not or does not fully demonstrate evidence on the following:   * Use of OO principles, i.e., encapsulation and abstraction. * Use of intermediate variables, constants, and try-catch blocks. * Idiomatic use of control flow, data structures and in-built functions. * Efficient algorithmic approach. * Sufficient modularity. * Commenting and formatting. * No dead or unused code. |
| **Documentation & Git Usage** | README file contains comprehensive evidence on the following:   * The app’s class diagram. * How to run the unit tests. * Known bugs if applicable.   Git commit messages are comprehensively formatted and reflect the changes in concise detail. | README file contains clear evidence of:   * The app’s class diagram. * How to run the unit tests. * Known bugs if applicable.   Git commit messages are clearly formatted and reflect the changes in substantial detail. | README file contains evidence of:   * The app’s class diagram. * How to run the unit tests. * Known bugs if applicable.   Git commit messages are formatted and reflect the changes in detail. | README file does not or does not fully contain evidence of:   * The app’s class diagram. * How to run the unit tests. * Known bugs if applicable.   Git commit messages are not or are not fully formatted and do not or do not fully reflect the changes. |

# **ID511001: Programming 2**

# **Project 1 (C# Console App): Learner Gradebook Marking Cover Sheet**

Name:

Date:

Learner ID:

Assessor’s Name:

Assessor’s Signature:

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Out Of** | **Weighting** | **Final Result** |
| Functionality | 10 | 40 |  |
| Code Elegance | 10 | 45 |  |
| Documentation & Git Usage | 10 | 15 |  |
| **Final Result** | | | /100 |
| **This assessment is worth 25% of the final mark for the Programming 2 course.** | | | |

**Feedback:**

**Functionality:**

**Code Elegance:**

**Documentation & Git Usage:**