INF1002 Programming Fundamentals

Team Project Specification – Class Management System (CMS)

Table of Contents

1. Introduction

A **database** is a structured collection of data organized for efficient storage, retrieval, and manipulation. A database typically contains multiple **tables**; each table consists of **columns** and **rows** (records). The first column is often a unique **ID** identifying each record.

Example data table: `FruitPrice`

ID	Price	Origin
Apple	3.7	USA
Orange	3.6	China
Watermelon	5.6	Malaysia

A **Database Management System (DBMS)** provides an interface for interacting with databases. Fundamental operations typically include **Insert (Add)**, **Query**, **Update (Edit)**, and **Delete**.

Your team will implement a simple command-line **Class Management System (CMS)** program in **C** that behaves like a miniature DBMS for a single table.

2. Detailed Requirements

2.1 File Database

Use a single file as the database to store records. Only one table, `StudentRecords`, is required, with the following columns:

Column	Data Type
ID	integer
Name	string
Programme	string
Mark	float

Sample records:

ID	Name	Programme	Mark
2301234	Joshua Chen	Software Engineering	70.5
2201234	Isaac Teo	Computer Science	63.4
2304567	John Levoy	Digital Supply Chain	85.9

Database filename convention: `TeamName-CMS.txt`
Example: `P1_1-CMS.txt`

A sample database file is provided: `Sample-CMS.txt`.

2.2 Operations

Your program must accept operation commands from the user and respond accordingly. See Appendix A for sample dialogs.

OPEN

- Open the database file and read in all records.

SHOW ALL

- Display all current records in memory (those read from the file and any unsaved changes).

INSERT

- Insert a new record.

- If a record with the same **student ID** already exists, display an error and cancel the insertion.
- If the ID is unique, prompt for each column value and add the record.

QUERY

- Search for a record by **student ID**.
- If found, display the record; otherwise, show a warning that no record exists.

UPDATE

- Update the record with a given **student ID**.
- If no matching ID exists, show a warning.
- If found, prompt for the new values for each column and update the record.

DELETE

- Delete the record with a given **student ID**.
- If no matching ID exists, show a warning.
- If found, **double-confirm** with the user. Proceed with deletion only if confirmed; otherwise cancel.

SAVE

- Save all current records back to the database file.

---

3. Enhancement Features

3.1 Sorting Features

- Implement sorting of student records by:
- **Student ID** (ascending/descending)
- **Mark** (ascending/descending)
- Sorted output should work with commands such as:
- `SHOW ALL SORT BY ID`
- `SHOW ALL SORT BY MARK`

3.2 Summary Statistics

- Implement summary commands, e.g., `SHOW SUMMARY`, that display:
- Total number of students
- Average mark
- Highest and lowest mark **(with student names)**

3.3 Unique Feature

- Propose **one unique feature** that:
- Demonstrates multiple features of C, and
- Fits naturally into the CMS system.

4. Test Case Design and Validation

Design a comprehensive suite of test cases to ensure correctness and robustness. Identify potential failure points and create tests for them. Use the table format below in your report and in your demo.

Sample Test Case Format

Test Case ID	Description	Input Command(s)	Expected Output	Reason for Test	Actual Result
TC05	Update Record Successfully	`UPDATE ID=250001` <br> `Mark=88.0`	`CMS: The record with ID=250001 is successfully updated`	Verifies Update Logic	Pass

Test cases should be used during development, listed in the report, and executed in your video presentation.

5. Declaration of Non-Plagiarism

When the CMS program starts, it **must** print the following declaration:

Declaration
SIT's policy on copying does not allow the students to copy source code as well as assessment solutions from another person, AI, or other places. It is the students' responsibility to guarantee that their assessment solutions are their own work. Meanwhile, the students must also ensure that their work is not accessible by others. Where such plagiarism is detected, both of the assessments involved will receive **ZERO** mark.

We hereby declare that: - We fully understand and agree to the abovementioned plagiarism policy. - We did not copy any code from others or from other places. - We did not share our codes with others or upload to any other places for public access and will not do that in the future. - We agree that our project will receive Zero mark if there is any plagiarism detected. - We agree that we will not disclose any information or material of the group project to others or upload to any other places for public access. - We agree that we did not copy any code directly from AI generated sources.

Declared by: Group Name (insert your group name)
Team members:
XXX
XXX
XXX
XXX

XXX
Date: (insert the date when you submit your group project)

---

6. Report Requirements

Alongside your code, submit a brief technical report explaining how the program works and how it was constructed. Include:

1. A technical description of the program structure.
2. The data structures used (if any) and the rationale for their selection.
3. A breakdown of each team member's contributions.
4. A complete list of test cases and validations (see Section 4).

The report should be detailed enough that a peer who has taken **INF1002** could implement a similar program without seeing your code. A user guide is **not** required.

Formatting: - $\leq$ 15 pages (excluding title page, table of contents, appendix) - Times New Roman, font size **11** or larger - Standard **margins and borders** - The report should be standalone (appendix is optional and for reference only)

---

7. Timeline and Deliverables

Final Submission — By 23:59 on Tuesday, 25 November 2025

Submit a **zip** file to your group's xSiTe Dropbox including: - All **C source code** - The compiled and runnable **.exe** file - **Report / Reflection** - **Presentation slides**

Filename format: `INF1002C-TeamX.zip`
Example: `INF1002C-P1_1.zip`

For the video presentation, upload one video to YouTube or a similar platform (you may mark it as unlisted). Include the link in your report **and** inside the zip. Ensure your instructor has access **before** submitting.

Peer Evaluation — By 23:59 on Wednesday, 26 November 2025

Complete your peer evaluation on **TEAMMATES**.

- Otherwise, a **10% penalty** applies to your own peer evaluation mark, and
- Contribution credits will be evenly distributed among the other team members.

Late Submission Policy

- **20% per day** penalty for late submission unless an extension is granted in advance.
- Extensions are considered **case-by-case**.

• Submissions more than **4 days** late will **not** be accepted and will receive **no mark**.

---

8. Assessment Criteria

Your assignment will be assessed according to the following criteria.

8.1 Code (80%)

Code Completeness & Correctness (45%) - All requirements implemented - Correct results and expected behaviours - Appropriate and efficient data structures

Code Clarity (5%) - Well-structured, modular code - Clear, sufficient comments - Meaningful names for variables and functions

Code Reliability (10%) - No bugs or errors - Sufficient data validation; boundary cases handled - Proper error messages for users

Enhancement Features (10%) - All enhancement features implemented successfully to achieve the highest grade

Test Cases and Validation (10%) - Comprehensive test cases cover each function and edge condition - Clear demonstration of coverage and correctness - Well-documented test case tables with expected and actual results (included in report)

8.2 Report (10%)

- Logical flow with all required sections (see Section 6)
- Clear elaboration of code structure and data structures
- Well organized, clear and concise writing
- Free from grammatical errors and typos
- Includes the full list of designed test cases

8.3 Individual Reflection (5%)

- Clear and honest account of your contributions
- Critical evaluation of the project (what went well / poorly)
- Honest reflection on how AI and an AI-teammate were used

8.4 Video Presentation & Demonstration (5%)

Presentation (max 5 minutes) - Functions/features, code structure, and data structures clearly presented - Time well managed

Video Demo (max 10 minutes) - All required functions demonstrated successfully - Comprehensive team test cases handled properly - Demo time well managed

Individual member grades are weighted by contribution and peer evaluation.

---

9. Individual Reflection & Presentation

• See Sections **8.3** and **8.4** for detailed expectations.

Appendix A. Sample Responses

Assume the program is named **CMS**, the user is **P1_1**, and the database file is `P1_1-CMS.txt`.

OPEN

```
P1_1: OPEN
CMS: The database file "P1_1-CMS.txt" is successfully opened.
```

SHOW ALL

```
P1_1: SHOW ALL
CMS: Here are all the records found in the table "StudentRecords".
ID      Name        Programme           Mark
2301234     Joshua Chen Software Engineering    70.5
2201234     Isaac Teo       Computer Science        63.4
2304567     John Levoy  Digital Supply Chain    85.9
```

INSERT

```
P1_1: INSERT ID=2301234
CMS: The record with ID=2301234 already exists.

P1_1: INSERT ID=2401234 Name=Michelle Lee Programme=Information Security
Mark=73.2
CMS: A new record with ID=2401234 is successfully inserted.

P1_1: SHOW ALL
CMS: Here are all the records found in the table "StudentRecords".
ID      Name        Programme           Mark
2301234     Joshua Chen Software Engineering    70.5
2201234     Isaac Teo       Computer Science        63.4
2304567     John Levoy  Digital Supply Chain    85.9
2401234     Michelle Lee    Information Security    73.2
```

QUERY

```
P1_1: QUERY ID=2501234
CMS: The record with ID=2501234 does not exist.

P1_1: QUERY ID=2401234
CMS: The record with ID=2401234 is found in the data table.
ID      Name        Programme           Mark
2401234     Michelle Lee    Information Security    73.2
```

UPDATE

```
P1_1: UPDATE ID=2501234 Mark=69.8
CMS: The record with ID=2501234 does not exist.

P1_1: UPDATE ID=2401234 Mark=69.8
CMS: The record with ID=2401234 is successfully updated.

P1_1: UPDATE ID=2401234 Programme=Applied AI
CMS: The record with ID=2401234 is successfully updated.

P1_1: SHOW ALL
CMS: Here are all the records found in the table "StudentRecords".
ID      Name        Programme       Mark
2301234     Joshua Chen Software Engineering    70.5
2201234     Isaac Teo       Computer Science    63.4
2304567     John Levoy  Digital Supply Chain    85.9
2401234     Michelle Lee    Applied AI      69.8
```

DELETE

```
P1_1: DELETE ID=2501234
CMS: The record with ID=2501234 does not exist.

P1_1: DELETE ID=2401234
CMS: Are you sure you want to delete record with ID=2401234? Type "Y" to confirm
or type "N" to cancel.

P1_1: N
CMS: The deletion is cancelled.

P1_1: DELETE ID=2301234
CMS: Are you sure you want to delete record with ID=2301234? Type "Y" to confirm
or type "N" to cancel.
```

```
P1_1: Y
CMS: The record with ID=2301234 is successfully deleted.

P1_1: SHOW ALL
CMS: Sure. Here are all the records found in the table "StudentRecords".
ID      Name        Programme           Mark
2201234     Isaac Teo       Computer Science        63.4
2304567     John Levoy  Digital Supply Chain    85.9
2401234     Michelle Lee    Applied AI      69.8
```

SAVE

```
P1_1: SAVE
CMS: The database file "P1_1-CMS.txt" is successfully saved.
```