Student Marks & Grade Management System

Project Report

C- Programming

Submitted by: Janvi Gupta

Course: B-Tech (HONS) (DATA ANALYTICS)

Year: 1st Year 2nd Semester

Table of Contents

1. Introduction

The **Student Marks & Grade Management System** is a C program designed to calculate and manage student grades based on marks obtained in five subjects. The program computes the total marks, average percentage, and assigns a grade (A, B, C, D, or F) based on predefined criteria.

This project demonstrates the use of **arrays, functions, conditional statements, and formatted output** in C programming. It serves as an efficient tool for educators to automate grade calculations.

2. Problem Statement

Manually calculating grades for multiple students is time-consuming and prone to errors. This program automates the process by:

- Accepting marks for five subjects.
- Calculating total and average marks.
- Assigning grades based on a predefined scale.
- Displaying results in a structured format.

3. Objectives

- Develop a **modular program** using functions.
- Implement **input validation** to ensure correct data entry.
- Use **arrays** to store subject marks efficiently.
- Apply **conditional statements** for grade determination.
- Ensure **user-friendly output formatting**.

4. System Design

4.1 Flowchart

Start → Input Marks → Validate Input → Calculate Total → Calculate Average → Determine Grade → Display Results → End

4.2 Functions Used

Function	Description
calculateTotal()	Computes sum of marks
calculateAverage()	Returns average percentage
determineGrade()	Assigns grade (A-F) based on average
displayResults()	Prints formatted results

4.3 Data Structures

- **Arrays** (marks [5]) store subject-wise marks.
- **Variables** (total, average, grade) hold computed results.

5. Implementation

5.1 Code Explanation

The program follows a **structured approach**:

1. **Input Phase:**
    a. Marks for 5 subjects are stored in an array.
    b. Input validation ensures marks are between 0-100.
2. **Processing Phase:**
    a. calculateTotal() sums up marks.
    b. calculateAverage() computes the mean.
    c. determineGrade() assigns a grade using if-else logic.
3. **Output Phase:**
    a. displayResults() shows total, average, and grade.

5.2 Input/Output Handling

- **Input:** printf("Enter marks for subject %d: ", i+1);
  scanf("%d", &marks[i]);

- **Output:** printf("Total Marks: %d\nAverage: %.2f\nGrade: %c\n", total, average, grade);

6. Testing & Results

6.1 Test Cases

Test Case	Input (Marks)	Expected Output
1	90, 92, 95, 88, 93	Total: 458, Avg: 91.6, Grade: A
2	75, 80, 72, 85, 78	Total: 390, Avg: 78.0, Grade: C
3	50, 55, 60, 45, 52	Total: 262, Avg: 52.4, Grade: F

6.2 Output Samples

Enter marks for subject 1: 85
Enter marks for subject 2: 92
...
Results:
Total Marks: 433
Average: 86.60
Grade: B

7. Conclusion

The program successfully automates grade calculation with:

✓ **Modularity** (using functions)

✓ **Efficiency** (arrays for storage)

✓ **User-friendly output**

Future Enhancements:

- Store multiple student records.
- Export results to a file.
- Add graphical interface.

8. References

1. Kernighan, B. W., & Ritchie, D. M. (1988). *The C Programming Language*.
2. "C Functions and Arrays" – GeeksforGeeks.

Appendix: Full Source Code (Attached)

This report provides a **comprehensive overview** of the project, covering design, implementation, and testing.