# Student Management System

1. **Introduction**

This Java-based console application is designed to manage basic academic data such as students, courses, subjects, and marks. It offers a modular and interactive environment where the user (typically an administrator or teacher) can perform CRUD (Create, Read, Update, Delete) operations on student and course data, assign marks for individual subjects, and generate grade reports.

The goal of the system is to offer a lightweight, efficient solution to manage educational records without the complexity of databases or external systems.

# Features

* 1. **Student Management**

This module allows the user to:

* + - Add new students by providing a unique student ID and a name.
    - Prevent duplication by checking if a student ID already exists.
    - View all registered students in the system in a clean tabular format.

# Course and Subject Management

This module facilitates course handling:

* + - Add new courses using a unique course code and a descriptive title.
    - View the list of available courses.
    - Each course can have one or more subjects added under it.
    - Subjects are managed within the scope of a course, allowing for curriculum-specific structures.

# Marks Management and Grading

The marks module provides the following capabilities:

* + - Assign marks to a student for a particular subject under a selected course.
    - Subjects are selected from the ones defined under the course.
    - The system automatically calculates the grade using a pre-defined grading scale.
    - View all marks assigned to students.
    - Search and filter results by student ID to view individual academic performance.

# Grading Criteria

Marks assigned are translated into grades according to the following scale:

* A+ : 90 and above
* A : 80 to 89
* B : 70 to 79
* C : 60 to 69
* D : 50 to 59
* F : Below 50

This makes it easier to assess performance quickly and categorize students based on merit.

# Module Description

* Student.java: Represents each student with an ID and name. Provides getter methods and a simple string representation.
* Course.java: Defines a course using a code and title. Maintains a list of subjects related to the course. Allows dynamic addition of subjects.
* Mark.java: Connects a student to a course and subject. Stores the score and computes the corresponding grade. Provides meaningful output.
* StudentManager.java: Acts as the main logic controller. Manages input, storage, and the user interface. Provides methods to manage students, courses, and marks.
* Manager.java: Contains the main method which triggers the application and initiates the menu-driven interface.

# Limitations and Future Improvements

* All data is stored in memory and is lost once the program terminates.
* There is no validation for duplicate subject entries under the same course.
* The user interface is text-based and can be improved with a graphical user interface (GUI) using Java Swing or JavaFX.
* There is no login or authentication system.
* No persistent storage - future enhancements can include file storage or integration with databases (e.g., SQLite, MySQL).
* Limited error handling in user inputs.
* Reports and statistical summaries could be added for deeper insights.

# Conclusion

The Student Management System is a structured, modular Java project suitable for beginners learning object-oriented programming and basic data handling. It lays the foundation for more advanced academic systems and can be expanded easily to support persistent data, reporting, authentication, and user roles.

This system is an ideal prototype or learning exercise for building educational management tools in Java.