Student Data Management System

A comprehensive Python & CSV solution for efficient academic record keeping and analysis.

Problem Statement

To develop a menu-driven Python program that utilizes the **CSV module** to create and manage a scalable database of student records. The system must efficiently perform read/write operations, specific record searching, and analytical comparisons of CGPA scores.

Top-Down Design

Data Management

Modules responsible for the core I/O operations: read_csv_file() for display and write_csv_file() for persistent storage.

Search & Retrieval

The search_csv_file() module allows efficient querying of records based on unique Admission Numbers.

Analytics

Specialized modules max_csv_file() and min_csv_file() to process numerical data and derive academic insights.

Algorithm: Write Function

- **Step 1:** Open Student_data.csv in 'append' or 'write' mode using the csv.writer object.
- **Step 2:** Initiate a loop to accept user inputs for fields: Admn_no, Name, Age, Grade, etc.
- **Step 3:** Store inputs in a list: data = [Admn_no, Name, ...].
- **Step 4:** Use writer.writerow(data) to save the record.
- **Step 5:** Prompt user to continue or break the loop.

Logical Flow

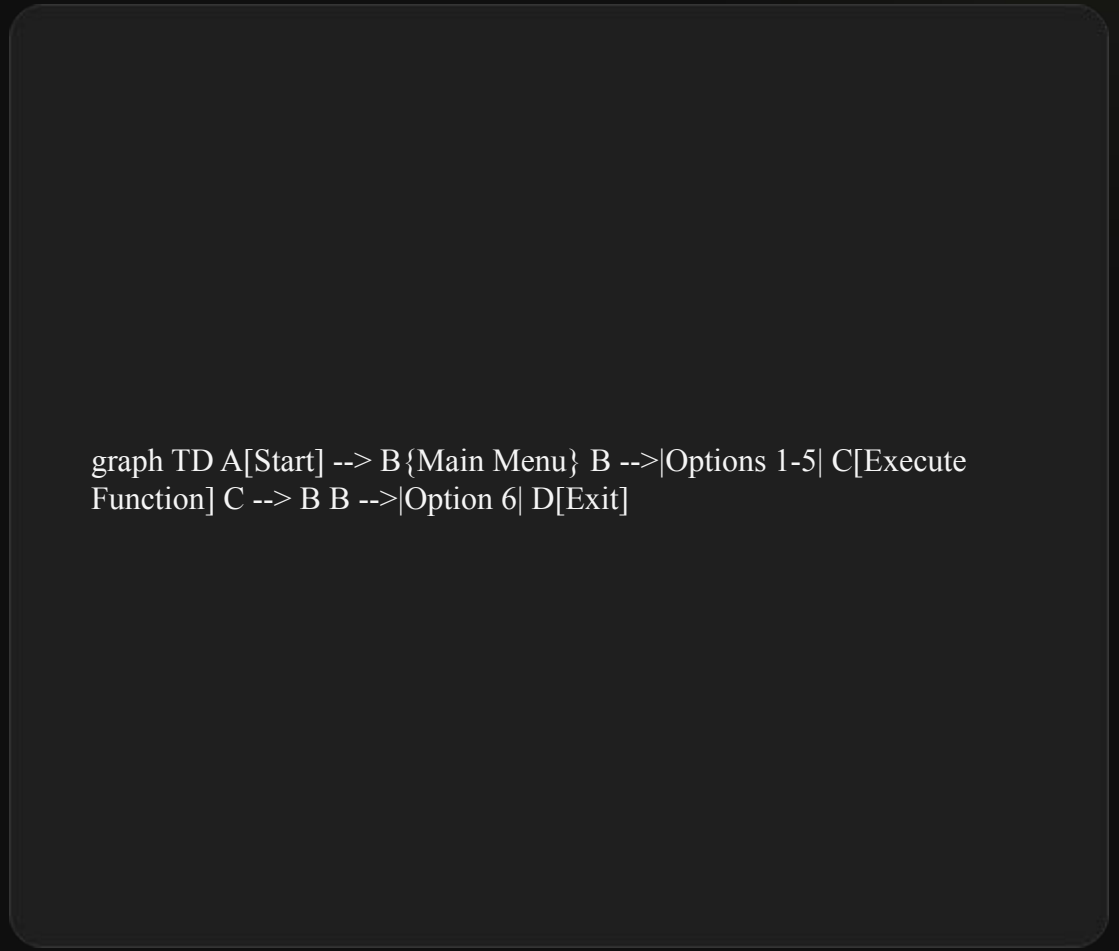
Start → Open File → Input Data → Write Row → Continue? → Stop

Program Overview

Simple Logic Cycle

The program operates on a continuous cycle until the user decides to exit. This ensures seamless workflow without restarting the application for every task.

- **Start:** Program initializes.
- **Menu:** Central hub for decision making.
- **Action:** User selection is executed.
- **Repeat:** Returns to menu automatically.



Code Implementation

Key Features

- **CSV Module:** Utilizes csv.reader and csv.writer for robust file handling.
- **Infinite Loop:** The main execution block runs inside a while True loop for continuous operation.
- **Error Handling:** Input validation (converting inputs to int or float) ensures data integrity.
- **Modularity:** Each distinct task is encapsulated in its own function for cleaner code.

Future Scope

GUI Integration

Enhance user experience by replacing the CLI with a graphical interface using **Tkinter** or **PyQt**.

SQL Database

Migrate from flat CSV files to a relational database system like **MySQL** or **SQLite** for better scalability.

Advanced Analytics

Integrate the **Pandas** and **Matplotlib** libraries to generate visual reports and complex data trends.

Thank You