🎓 Student Management System (CLI-based in Python with OOP)

🎯 Objective

Design and implement a Command Line Interface (CLI) based Student Management System

using Object-Oriented Programming (OOP) principles in Python. The system should be able to:

 Manage student records

 Handle course enrollments

 Assign and track grades

 Save/load data to/from a file (JSON format)

Section A: Class Design and Implementation

Class 1: Person

 Attributes:

o name (str): Name of the person.

o age (int): Age of the person.

o address (str): Address of the person.

 Method:

o display_person_info(): Prints the person’s details.

Class 2: Student (Inherits from Person)

 Additional Attributes:

o student_id (str): Unique identifier for each student.

o grades (dict): Subjects and respective grades (e.g., {"Math": "A"}).

o courses (list): List of enrolled courses.

 Methods:

o add_grade(subject, grade): Add or update the grade for a subject.

o enroll_course(course): Enroll in a specified course.

o display_student_info(): Print all student details.

Class 3: Course

 Attributes:

o course_name (str): Name of the course.

o course_code (str): Unique course code.

o instructor (str): Instructor’s name.

o students (list): Enrolled student objects.

 Methods:

o add_student(student): Add a student to the course.

o display_course_info(): Print course details and enrolled students.

Section B: System Functionalities

Implement a menu-driven CLI interface with the following options:

1. Add New Student

 Input: Name, Age, Address, Student ID

 Action: Create and store new Student object

 Output:

Student Sami (ID: S001) added successfully.

2. Add New Course

 Input: Course Name, Course Code, Instructor

 Action: Create and store new Course object

 Output:

Course Physics (Code: PHY101) created with instructor Dr. ABC.

3. Enroll Student in Course

 Input: Student ID, Course Code

 Action: Enroll student if both exist

 Output:

Student Sami (ID: S001) enrolled in Physics (Code: PHY101).

4. Add Grade for Student

 Input: Student ID, Course Code, Grade

 Action: Assign/update grade (only if enrolled)

 Output:

Grade A added for Sami in Physics.

5. Display Student Details

 Input: Student ID

 Output:

Student Information:

Name: Sami

ID: S001

Age: 22

Address: Dhaka

Enrolled Courses: Physics

Grades: {'Physics': 'A'}

6. Display Course Details

 Input: Course Code

 Output:

Course Information:

Course Name: Physics

Code: PHY101

Instructor: Dr. ABC

Enrolled Students: Sami

7. Save Data to File

 Function: save_data()

 Action: Save students and courses to a file (JSON)

 Output:

All student and course data saved successfully.

8. Load Data from File

 Function: load_data()

 Action: Restore data from file

 Output:

Data loaded successfully.

0. Exit

 Output:

Exiting Student Management System. Goodbye!

Section C: Error Handling & File Operations

Error Handling

 Validate student_id and course_code before any operation

 Ensure students are enrolled in a course before assigning grades

Bonus Challenge: File Operations

 Use json module for saving/loading data

 Must handle:

o Students and their attributes

o Courses and enrolled students

o Grades and enrollments

Sample Input / Output

==== Student Management System ====

1. Add New Student

2. Add New Course

3. Enroll Student in Course

4. Add Grade for Student

5. Display Student Details

6. Display Course Details

7. Save Data to File

8. Load Data from File

0. Exit

Select Option: 1

Enter Name: Sami

Enter Age: 22

Enter Address: Dhaka

Enter Student ID: S001

Student Sami (ID: S001) added successfully.