Overview
This educational management system is a console-based application designed to manage students, courses, and assignments.
It uses Object-Oriented Programming (OOP) principles to create a modular and scalable system.

Classes

1- Person

--Description: Base class for all users in the system.

-Attributes:
-name: The full name of the person.
-user_name: The username used for login.
-password: The password used for login.

2-Student

-- Description: Represents a student in the system.

-Attributes:
-id: The student ID.
-email: The email address of the student.
-registered_courses: List of courses the student is registered in.
-assignment_solutions: List of assignment solutions submitted by the student.

-Methods:
- register_course(course: Course): Registers the student in a given course and adds the student to the course's student list.
- list_courses() -> List[Course]: Returns a list of courses the student is registered in.
- view_course(course_code: str) -> Course : Returns the course object for a given course code if the student is registered in that course.
- submit_assignment(course_code: str, assignment_title: str, solution: str) -> AssignmentSolution : Submits an assignment solution for a given course and assignment.

3- Doctor

--Description: Represents a doctor (instructor) in the system.

-Attributes:
-teaching_courses: List of courses the doctor is teaching.

-Methods:
-create_course(name: str, code: str) -> Course: Creates a new course and adds it to the list of courses the doctor is teaching.
-list_courses() -> List[Course]: Returns a list of courses the doctor is teaching.
-view_course(course_code: str) -> Course: Returns the course object for a given course code if the doctor is teaching that course.
-create_assignment(course_code: str, title: str, description: str,

due_date: str) -> Assignment: Creates a new assignment for a given course.

4- Course

--Description: Represents a course in the system.

-Attributes:
-name: The name of the course.
-code: The course code.
-doctor: The doctor teaching the course.
-students: List of students registered in the course.
-assignments: List of assignments for the course.

-Methods:
-add_student(student: Student): Adds a student to the course.
-remove_student(student: Student): Removes a student from the course.
-add_assignment(title: str, description: str, due_date: str) -> Assignment: Adds an assignment to the course.
-remove_assignment(assignment: Assignment): Removes an assignment from the course.
-list_assignments() -> List[Assignment]: Returns a list of assignments for the course.

5- Assignment

--Description: Represents an assignment in a course.

-Attributes:
-title: The title of the assignment.
-description: The description of the assignment.
-due_date: The due date of the assignment.
-course: The course to which the assignment belongs.
-submissions: List of assignment solutions submitted by students.

-Methods:
-add_solution(solution: AssignmentSolution): Adds a solution to the list of submissions.
-list_solutions() -> List[AssignmentSolution]: Returns a list of solutions submitted for the assignment.

6-AssignmentSolution

--Description: Represents a solution submitted by a student for an assignment.

-Attributes:
-student: The student who submitted the solution.
-assignment: The assignment for which the solution was submitted.
-submission_date: The date of submission.
-grade: The grade received for the solution.
-comments: Comments on the solution.
-solution: The solution content.

-Methods:
-get_grade() -> Optional[str]: Returns the grade for the solution.
-set_grade(grade: str): Sets the grade for the solution.
-set_comment(comment: str): Sets the comment for the solution.

---Implementation Details

-Main Program

The main program provides a console-based user interface for interacting with the system. It includes functionality for:
Signing in and signing up for both students and doctors.
Viewing and managing courses and assignments.
Registering for courses and submitting assignment solutions.
The program initializes with some predefined data for students, doctors, and courses, making it easier to test and demonstrate the functionality.

Data Initialization

# RUNNING THE SYSTEM

القائمة الرئيسية

```
Welcome to the Educational Management System
1. Sign In
2. Sign Up
3. Exit
Choose an option:
```

تسجيل الدخول كدكتور
اختيار "1" ثم إدخال بيانات الدكتور:

```
Username: ali
Password: pass
```

قائمة الدكتور

```
Welcome Dr. Ali

1. List Courses
2. Create Course
3. View Course
4. Log out
Choose an option:
```

إنشاء كورس جديدة
اختيار "2"

```
Course Name: Programming 101
Course Code: CS101
Course created successfully!
```

################################

قائمة الطالب
تسجيل الدخول كطالب:

```
Welcome Hussien Samy

1. Register in Course
2. List My Courses
3. View a Course
4. Grades Report
5. Log out
Choose an option:
```

تسجيل في كورس
اختيار "1"

```
Available Courses:
1. Prog 1 (CS111)
2. Prog 2 (CS112)
3. Math 1 (CS123)
4. Math 2 (CS333)
5. Prog 3 (CS136)
6. Stat 1 (CS240)
7. Stat 2 (CS350)
Choose a course to register in: 1
Registered successfully!
```

عرض الكورسات الخاصة بالطالب
اختيار "2"

```
My Courses:
Prog 1 (CS111)
```

عرض كورس معينة
اختيار "3"

```
Enter Course Code: CS111

Course: Prog 1 (CS111)
Assignments:
No assignments found.

1. Submit Assignment Solution
2. Unregister from Course
3. Back
Choose an option:
```

يبدأ البرنامج بعرض القائمة الرئيسية.
يمكن للمستخدم تسجيل الدخول كدكتور أو طالب.
الدكتور يمكنه إنشاء كورسات وعرضها.
الطالب يمكنه تسجيل في كورسات، عرض كورساته، وعرض تفاصيل الكورسات.