**Student Management System Documentation**

**Problem Statement:**

Managing student data, course enrollments, and exam performance manually can be both time-consuming and inefficient for educational institutions. Students often face difficulties in accessing their courses and tracking their progress, while admins lack proper tools to manage course content and results. To address these issues, this project aims to create a console-based Student Management System using Java, with separate roles for students and admins. The system simplifies the learning process for students and provides essential administrative functions to manage everything in one place.

**Project Objectives**

* Build a console-based application in Java that follows Object-Oriented Programming (OOP) principles.
* Allow students to register, log in, enroll in courses, attempt exams, and view their results.
* Provide admin functionality to manage courses, view students, and monitor results.
* Ensure the code is clean, modular, and easy to extend using abstraction, inheritance, encapsulation, and polymorphism.

**System Modules Overview**

1. User Module (Abstract Class)

* Acts as a common parent for both Admin and Student.
* Defines a shared interface for showing the menu, which is implemented differently in each role.

1. Admin Module

The admin can:

* Add new courses, each with 5 multiple-choice questions (MCQs)
* Delete existing courses
* View all registered students with their details
* See the exam results of all students
* Login is done using hardcoded credentials (admin/admin123).

1. Student Module

Students can:

* Register and log in
* Enroll in available courses (one at a time)
* Take an exam consisting of MCQs
* Review their past results
* Use Training Mode to practice questions with hints
* Track progress in training mode

1. Course Module

Each course consists of:

* A course name
* A set of 5 pre-defined MCQs

1. Question Module

Each question includes:

* The question text
* Four options (a–d)
* The correct option (stored as a lowercase string like "a", "b", etc.)

1. ExamResult Module

* Stores the outcome of exams, including:
* Course name
* Score percentage
* Pass/Fail status (Students must score 50% or above to pass)

1. Main System (StudentManagementSystem)

Handles:

* Student registration and login
* Admin login
* Maintains course and student lists
* Runs the application from the main() method

**Technologies Used**

* Language: Java
* Key Concepts: OOP (Abstraction, Inheritance, Encapsulation, Polymorphism)
* Data Structures: ArrayList, HashMap
* User Input: Scanner
* Exception Handling: try-catch blocks for input safety

**Key Features**

* Separate roles for admin and student
* Secure student registration and login
* Admin can add/delete courses with custom MCQs
* Students can enroll in one course, attempt exams, and get instant results
* Training Mode allows students to practice with hints and retry incorrect answers
* Tracks and displays training progress
* Admin can view all students and their performances
* Two sample courses are preloaded:

1. Data Structures
2. Operating Systems

**How the System Works**

* The system starts and displays the main menu.
* The admin logs in using default credentials to manage courses and students.
* Students can register and then log in with their credentials.
* The admin can add new courses or remove existing ones.

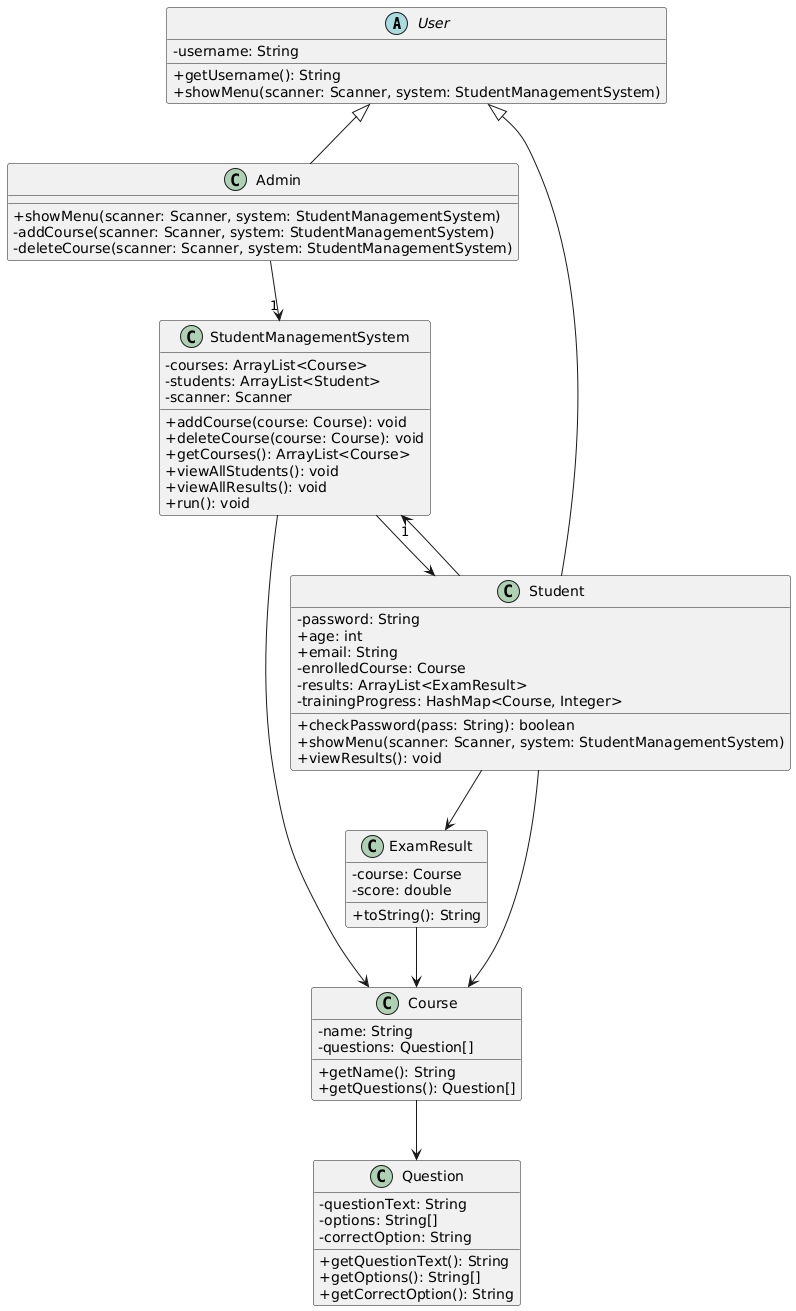
Students can:

* Enroll in a course
* Attempt the exam (score ≥50% to pass)
* Use training mode for learning and hints
* Track training progress and view results

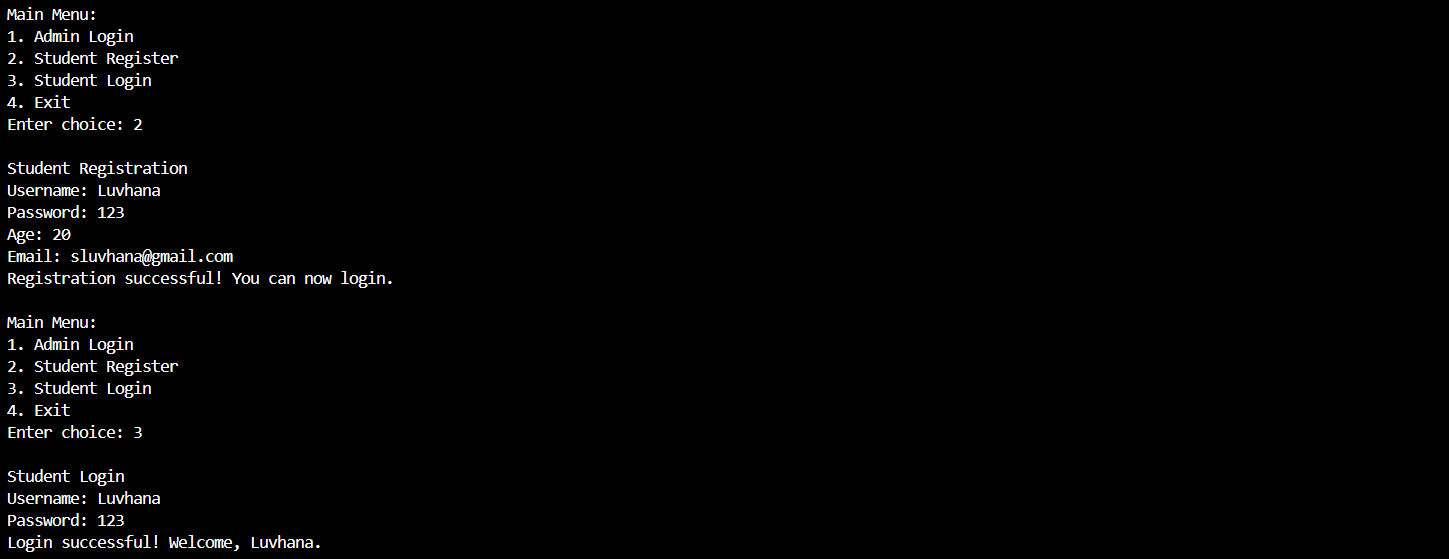
The admin can view:

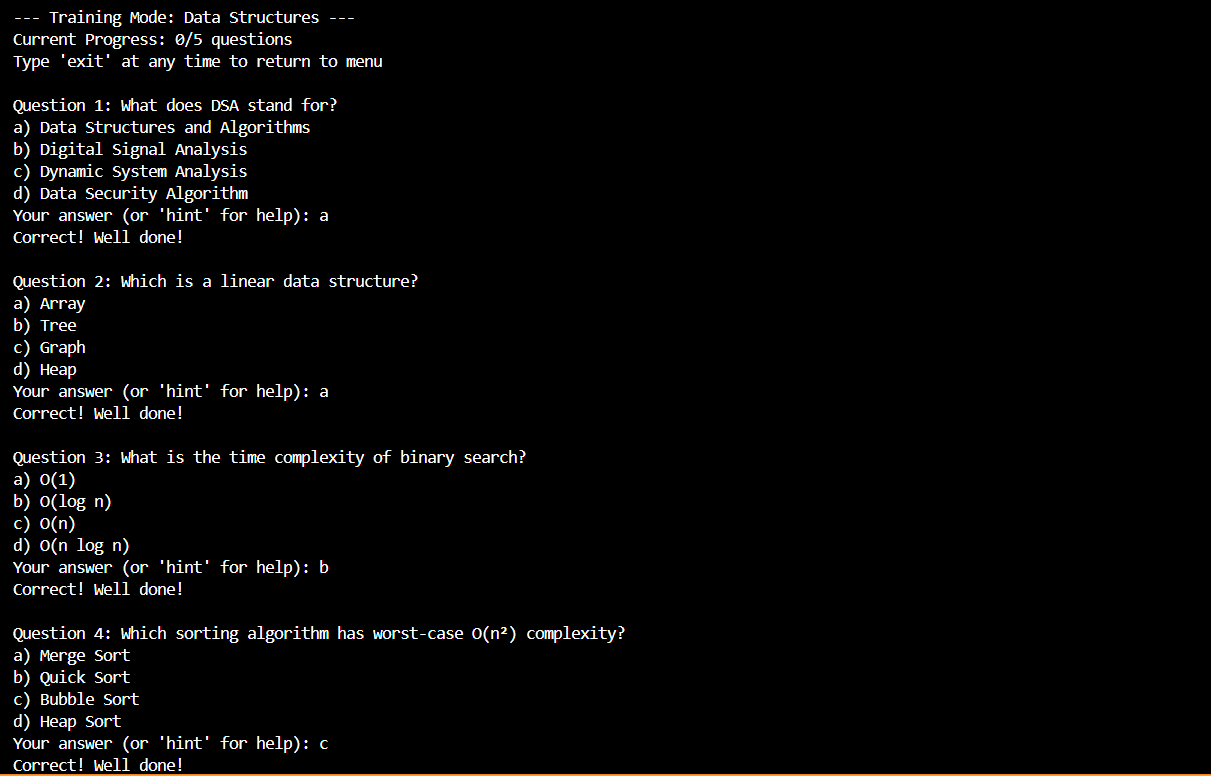
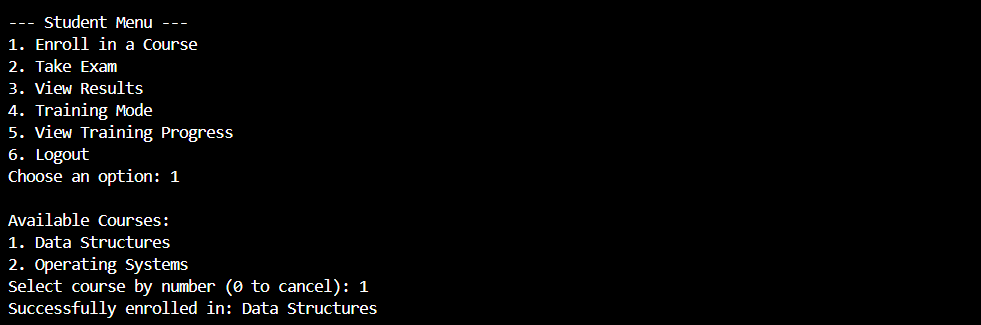
* All registered students with details
* Exam results of each student

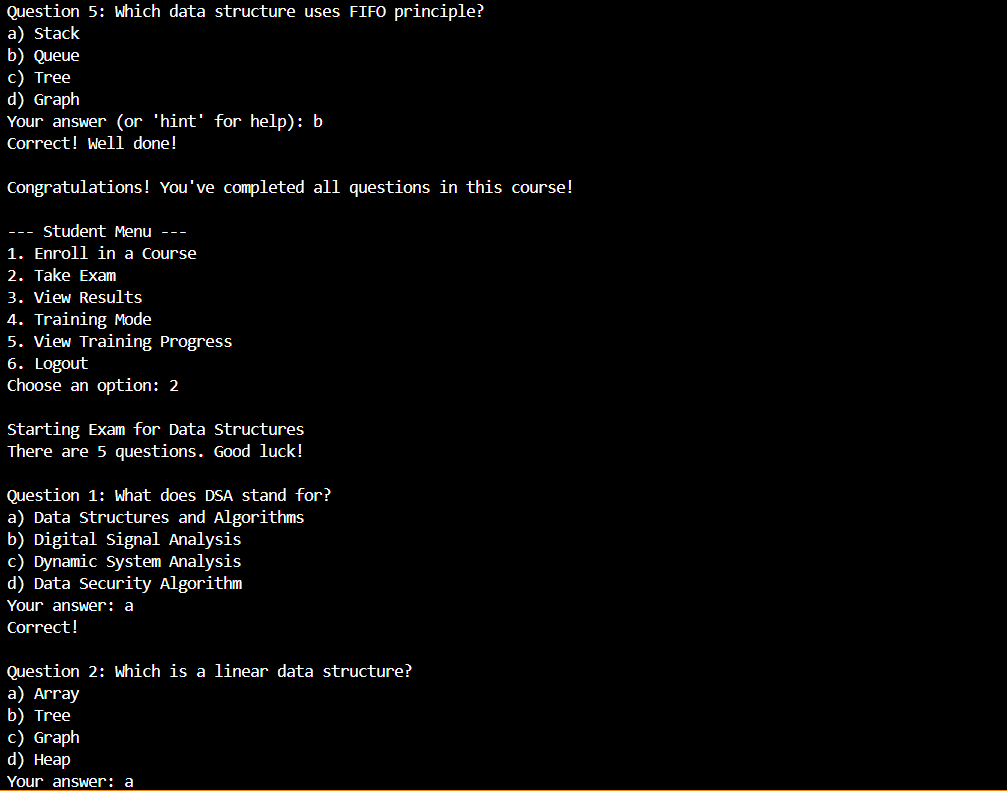
**Class Diagram**

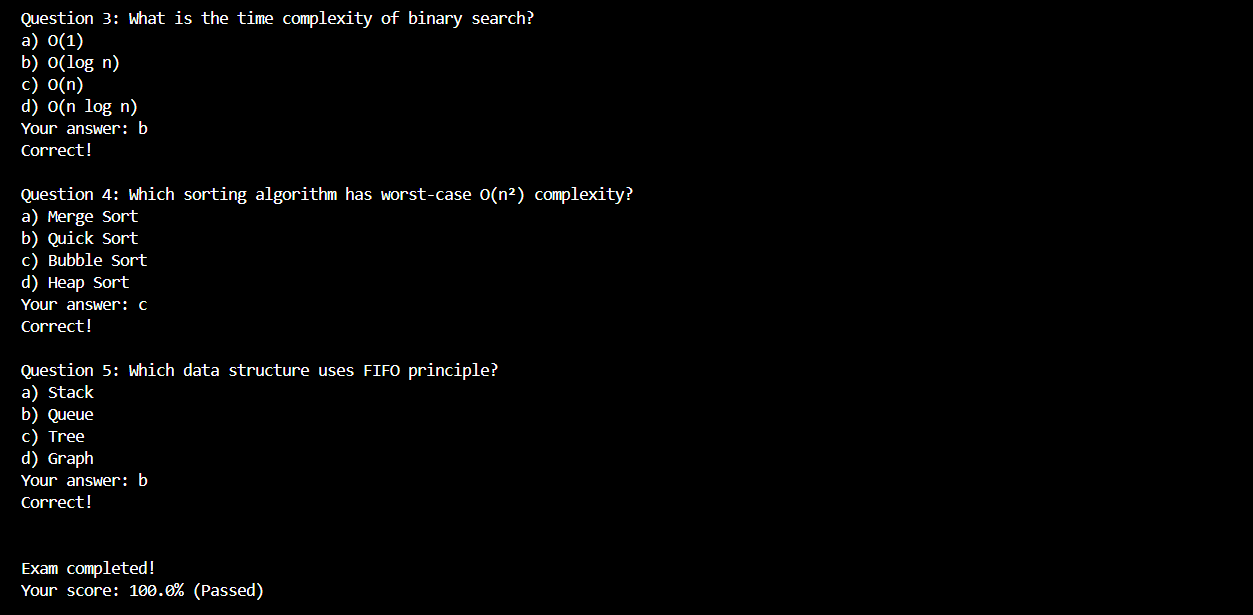


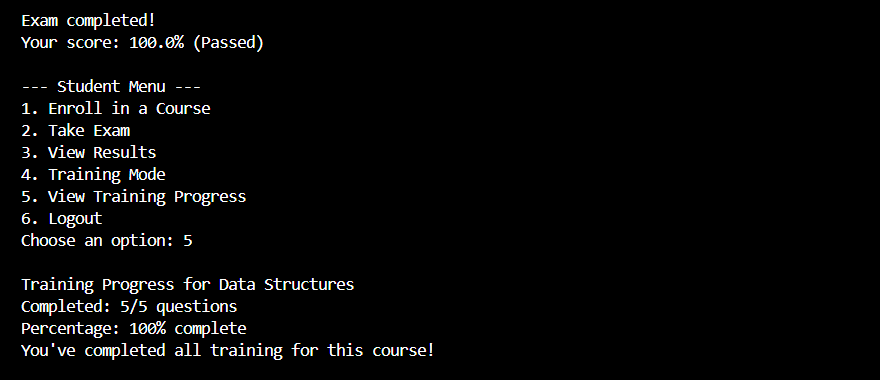
**Screenshots**

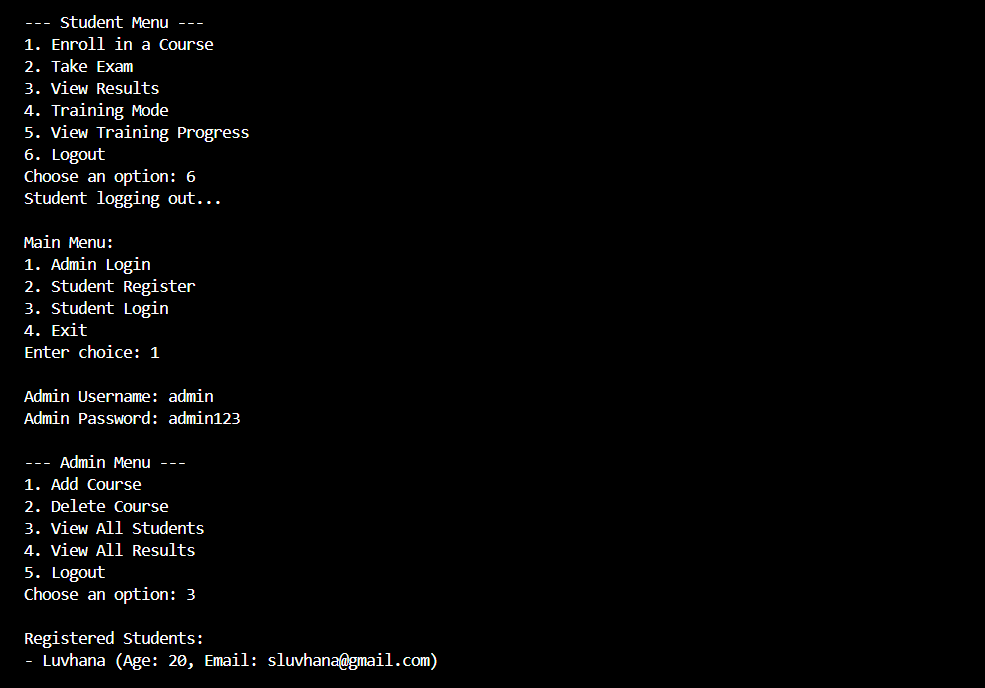


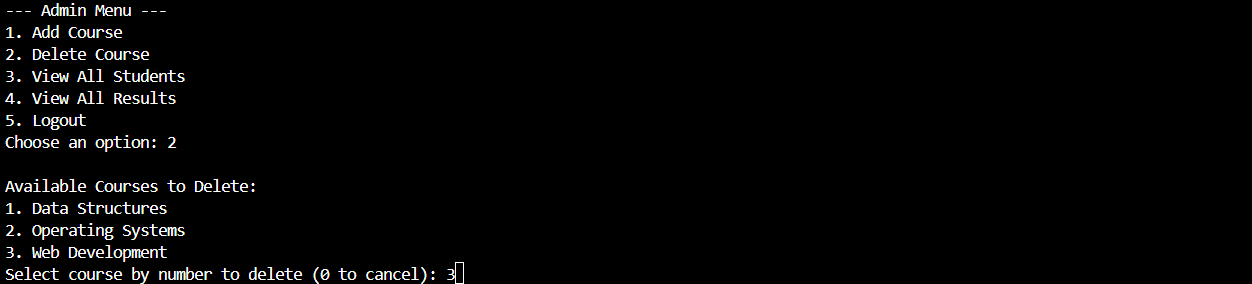












**Conclusion**

This Student Management System provides a simple yet effective solution to manage educational content and student progress in a structured way. Built with Java and Object-Oriented Programming, it keeps the code clean, modular, and easy to extend.The Training Mode, exam system, and admin controls make it a great foundation for more advanced systems in the future. With enhancements like a GUI, database integration, or support for multiple course enrollments, this project can easily scale to meet more complex academic needs.