1. **Java Code – Student Management System**

import java.util.ArrayList;

import java.util.Scanner;

// Student class

class Student {

private int id;

private String name;

private String grade;

// Constructor

public Student(int id, String name, String grade) {

this.id = id;

this.name = name;

this.grade = grade;

}

// Getters

public int getId() {

return id;

}

// toString method for display

@Override

public String toString() {

return "Student ID: " + id + ", Name: " + name + ", Grade: " + grade;

}

}

// Main class

public class StudentManagementSystem {

private ArrayList<Student> students;

private Scanner sc;

public StudentManagementSystem() {

students = new ArrayList<>();

sc = new Scanner(System.in);

}

// Add a student

public void addStudent() {

System.out.print("Enter Student ID: ");

int id = sc.nextInt();

sc.nextLine(); // consume newline

System.out.print("Enter Student Name: ");

String name = sc.nextLine();

System.out.print("Enter Student Grade: ");

String grade = sc.nextLine();

students.add(new Student(id, name, grade));

System.out.println("✅ Student added successfully!");

}

// Remove student by ID

public void removeStudent() {

System.out.print("Enter Student ID to remove: ");

int id = sc.nextInt();

boolean removed = students.removeIf(s -> s.getId() == id);

if (removed) {

System.out.println("✅ Student removed successfully!");

} else {

System.out.println("⚠ No student found with that ID.");

}

}

// Display all students

public void displayStudents() {

if (students.isEmpty()) {

System.out.println("⚠ No students to display.");

return;

}

System.out.println("\n📋 Student List:");

for (Student s : students) {

System.out.println(s);

}

}

// Menu

public void menu() {

while (true) {

System.out.println("\n--- Student Management System ---");

System.out.println("1. Add Student");

System.out.println("2. Remove Student");

System.out.println("3. Display Students");

System.out.println("4. Exit");

System.out.print("Choose an option: ");

int choice = sc.nextInt();

switch (choice) {

case 1: addStudent(); break;

case 2: removeStudent(); break;

case 3: displayStudents(); break;

case 4:

System.out.println("👋 Exiting... Goodbye!");

return;

default:

System.out.println("⚠ Invalid choice. Try again.");

}

}

}

// Main method

public static void main(String[] args) {

StudentManagementSystem sms = new StudentManagementSystem();

sms.menu();

}

}

1. **What is a class in Java?**  
   A class is a blueprint for creating objects, defining their properties (fields) and behaviors (methods).
2. **Difference between a class and an object?**
   * Class: Template/blueprint.
   * Object: An instance of a class created in memory.
3. **How does ArrayList differ from an array?**
   * Array: Fixed size, stores primitive types or objects.
   * ArrayList: Resizable, stores only objects, provides built-in methods for manipulation.
4. **What is encapsulation in Java?**  
   Wrapping data (fields) and methods together while restricting direct access to fields using access modifiers.
5. **How do you create and use constructors?**  
   Constructors are special methods with the same name as the class, used to initialize objects.
6. public Student(int id, String name) { ... }
7. Student s = new Student(1, "John");
8. **What is method overloading?**  
   Defining multiple methods with the same name but different parameter lists in the same class.
9. **How does remove() method work in ArrayList?**
   * remove(index) → removes element at a specific position.
   * remove(object) → removes the first occurrence of the object.
10. **Purpose of toString() method?**  
    Provides a readable string representation of an object, useful for debugging and display.
11. **Why use ArrayList instead of arrays?**  
    It’s dynamic in size, easier to insert/remove elements, and comes with built-in utility methods.
12. **Difference between ArrayList and LinkedList?**
    * ArrayList: Backed by an array, faster for random access, slower for insert/remove in the middle.
    * LinkedList: Doubly linked list, faster for frequent insert/remove, slower for random access.

**3. Suggested README.md for GitHub**

# Student Management System – Java OOP

## 📌 Overview

A simple Java program to manage students using Object-Oriented Programming concepts.

Features:

- Add a new student

- Remove student by ID

- Display all students

## 🛠 Tech Stack

- Java

- ArrayList

- OOP (Encapsulation, Constructors, toString)

## 🚀 How to Run

1. Clone the repo

2. Open in Eclipse, IntelliJ, or VS Code

3. Compile and run `StudentManagementSystem.java`

## 📖 Sample Run

--- Student Management System ---

1. Add Student
2. Remove Student
3. Display Students
4. Exit  
   Choose an option: 1  
   Enter Student ID: 101  
   Enter Student Name: John  
   Enter Student Grade: A  
   ✅ Student added successfully!

## 🎯 Learning Outcomes

- Java classes & objects

- Constructors & methods

- Using ArrayList for dynamic storage

- Menu-driven program

## 💡 Interview Prep Q&A

1. What is a class in Java? – \*A blueprint for creating objects…\*

2. Difference between class & object – \*Class is a template, object is an instance…\*

3. Array vs ArrayList – \*Array is fixed size, ArrayList is dynamic…\*

4. Encapsulation – \*Wrapping data & methods…\*

5. Constructors – \*Special method to initialize…\*

6. Method overloading – \*Same method name, different params…\*

7. ArrayList remove() – \*Removes by index or object…\*

8. toString() – \*Readable object string…\*

9. Why ArrayList – \*Dynamic, easy to manage…\*

10. ArrayList vs LinkedList – \*ArrayList fast random access, LinkedList fast inserts/removes…\*