Problem Statement: This program demonstrates the basics of object-oriented programming using a simple Student Record System. It shows how classes, objects, methods, and encapsulation work together to store and manage student details such as roll number, name, course, marks, and grade. The program uses methods for input, grade calculation, and displaying information. A menu-driven approach allows adding and viewing student records. Overall, it introduces foundational OOP concepts and basic input handling in Java.

1. Purpose of the Program

The aim of this program is to create a simple Student Record System where the user can enter details of a student such as roll number, name, course, and marks. The program then calculates the grade based on the marks and displays the complete student information.

2. Use of Class and Object

A class named Student is created to represent a student.
It contains:

- Data members → rollNo, name, course, marks, grade
- Methods → inputDetails(), calculateGrade(), displayDetails()

Objects of this class are used to store and manage each student's information.

3. Use of Methods

- inputDetails()
  Takes user input (roll no, name, course, marks).
- calculateGrade()
  Calculates grade based on the marks:

○ A → 90 and above

○ B → 75–89 ○ C →

50–74 ○ D → below

50

- displayDetails()
  Prints all the student information on the screen.

4. Menu-Driven Program

The program uses a loop and a menu to perform different operations:

1. Add Student → Creates and stores a new student record.

2. Display Students → Shows details of all added students.
3. Exit → Ends the program safely.

This allows the user to interact with the system easily.

5. Use of ArrayList

All student objects are stored in an ArrayList so that:

- Multiple students can be added
- Records can grow dynamically
- Easy traversal and display of records become possible

6. Calculating Grade

Based on the marks entered by the user, the program assigns a grade using simple if–else conditions.

This helps in evaluating the student's performance.

7. Scanner Input Handling

The Scanner class is used to take input from the user.

Input for roll number, name, course, and marks is handled properly, and the scanner is closed safely when the program exits.

8. Output Display

The program prints all student details in a clean format so the user can clearly understand the output.

It shows roll numbers, name, course, marks, and the calculated grade.

9. Overall Functionality

The program demonstrates basic Object-Oriented Programming (OOP) concepts such as:

- Class
- Object
- Methods
- Encapsulation

It also uses Java's Collections Framework (ArrayList) and interactive menu handling to create a simple student management system.

Input:

```java
import java.util.*;

class Student {
    int rollNo;
    String name;
    String course;
    double marks;
    char grade;

    void inputDetails() {
        Scanner sc = new Scanner(System.in);

        System.out.print(s: "Enter Roll Number: ");
        rollNo = sc.nextInt();
        sc.nextLine();

        System.out.print(s: "Enter Name: ");
        name = sc.nextLine();

        System.out.print(s: "Enter Course: ");
        course = sc.nextLine();

        System.out.print(s: "Enter Marks: ");
        marks = sc.nextDouble();

        calculateGrade();
    }

    void calculateGrade() {
        if (marks >= 90) grade = 'A';
        else if (marks >= 75) grade = 'B';
        else if (marks >= 50) grade = 'C';
```

```
33          else grade = 'D';
34      }
35
36      void displayDetails() {
37          System.out.println(x: "\n--- Student Details ---");
38          System.out.println("Roll No: " + rollNo);
39          System.out.println("Name: " + name);
40          System.out.println("Course: " + course);
41          System.out.println("Marks: " + marks);
42          System.out.println("Grade: " + grade);
43      }
44  }
45
46  public class Assignment1 {
47
        Run | Debug
48      public static void main(String[] args) {
49
50          Scanner sc = new Scanner(System.in);
51          ArrayList<Student> list = new ArrayList<>();
52
53          while (true) {
54              System.out.println(x: "\n--- MENU ---");
55              System.out.println(x: "1. Add Student");
56              System.out.println(x: "2. Display All Students");
57              System.out.println(x: "3. Exit");
58              System.out.print(s: "Enter your choice: ");
59
60              int choice = sc.nextInt();
61
```

```
62                  switch (choice) {
63                      case 1:
64                          Student s = new Student();
65                          s.inputDetails();
66                          list.add(s);
67                          break;
68
69                      case 2:
70                          if (list.isEmpty()) {
71                              System.out.println(x: "No students added yet.");
72                          } else {
73                              for (Student stu : list) {
74                                  stu.displayDetails();
75                              }
76                          }
77                          break;
78
79                      case 3:
80                          System.out.println(x: "Exiting program...");
81                          sc.close();
82                          return;
83
84                      default:
85                          System.out.println(x: "Invalid choice! Try again.");
86                  }
87              }
88          }
89      }
90
```

Output:

```
--- MENU ---
1. Add Student
2. Display All Students
3. Exit
Enter your choice: 1
Enter Roll Number: 102
Enter Name: Khushi
Enter Course: bca
Enter Marks: 99

--- MENU ---
1. Add Student
2. Display All Students
3. Exit
Enter your choice: 2

--- Student Details ---
Roll No: 102
Name: Khushi
Course: bca
Marks: 99.0
Grade: A

--- MENU ---
1. Add Student
2. Display All Students
3. Exit
Enter your choice: 3
Exiting program...
PS C:\Users\Hp\JAVA1>
```