OVERALL POST ASSESSMENT TASK

1)DAY-1:
Write a program to accept student marks and print results using the grading logic and functions

```
import java.util.Scanner;

public class StudentGrade {

    // Function to determine grade
    static String getGrade(double average) {
        if (average >= 90) return "A";
        else if (average >= 80) return "B";
        else if (average >= 70) return "C";
        else if (average >= 60) return "D";
        else return "F (Fail)";
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter marks for Subject 1: ");
        int m1 = sc.nextInt();
        System.out.print("Enter marks for Subject 2: ");
        int m2 = sc.nextInt();
        System.out.print("Enter marks for Subject 3: ");
        int m3 = sc.nextInt();

        int total = m1 + m2 + m3;
        double average = total / 3.0;

        System.out.println("\nTotal Marks: " + total);
        System.out.printf("Average: %.2f\n", average);
        System.out.println("Grade: " + getGrade(average));

        sc.close();
    }
}
```

Output:

```
Enter marks for Subject 1: 85
Enter marks for Subject 2: 78
Enter marks for Subject 3: 92
```

Total Marks: 255
Average: 85.00
Grade: B

2)DAY-3
Create a Library class system: add, remove, and issue books

```
import java.util.ArrayList;

public class SimpleLibrary {
    public static void main(String[] args) {
        ArrayList<String> books = new ArrayList<>();

        // Add books
        books.add("Java Programming");
        books.add("Python Basics");
        books.add("C++ for Beginners");

        // Display books
        System.out.println("Books in Library:");
        for (String book : books) {
            System.out.println("- " + book);
        }

        // Issue a book
        String bookToIssue = "Python Basics";
        if (books.contains(bookToIssue)) {
            books.remove(bookToIssue);
            System.out.println("\nBook issued: " + bookToIssue);
        }

        // Display books after issuing
        System.out.println("\nBooks left in Library:");
        for (String book : books) {
            System.out.println("- " + book);
        }
    }
}
```

Output:

```
Books in Library:
- Java Programming
- Python Basics
- C++ for Beginners

Book issued: Python Basics
```

Books left in Library:
- Java Programming
- C++ for Beginners

3)DAY-4

Develop a file-based Employee record system with CRUD using files.

```
import java.io.*;
import java.util.Scanner;

public class SimpleEmployeeSystem {
    static final String FILE_NAME = "employees.txt";

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        while (true) {
            System.out.println("\n1. Add Employee");
            System.out.println("2. View Employees");
            System.out.println("3. Exit");
            System.out.print("Choose option: ");
            int choice = sc.nextInt();

            switch (choice) {
                case 1 -> addEmployee(sc);
                case 2 -> viewEmployees();
                case 3 -> {
                    System.out.println("Goodbye!");
                    return;
                }
                default -> System.out.println("Invalid choice.");
            }
        }
    }

    static void addEmployee(Scanner sc) {
        try {
            sc.nextLine(); // consume leftover newline
            System.out.print("Enter ID: ");
            String id = sc.nextLine();
            System.out.print("Enter Name: ");
            String name = sc.nextLine();

            FileWriter fw = new FileWriter(FILE_NAME, true);
            fw.write(id + "," + name + "\n");
            fw.close();
```

```
            System.out.println("Employee added.");
        } catch (IOException e) {
            System.out.println("Error writing to file.");
        }
    }

    static void viewEmployees() {
        try {
            BufferedReader br = new BufferedReader(new FileReader(FILE_NAME));
            String line;
            System.out.println("\n--- Employee List ---");
            while ((line = br.readLine()) != null) {
                String[] data = line.split(",");
                System.out.println("ID: " + data[0] + ", Name: " + data[1]);
            }
            br.close();
        } catch (IOException e) {
            System.out.println("No employees found.");
        }
    }
}
```

Output:

```
1. Add Employee
2. View Employees
3. Exit
Choose option: 1
Enter ID: 101
Enter Name: Anushika
Employee added.

Choose option: 2
--- Employee List ---
ID: 101, Name: Anushika
```

4)DAY-8

Create Employee and Department tables and insert records

```
MariaDB [sd]> CREATE TABLE Department (
    ->     DeptID INT PRIMARY KEY,
    ->     DeptName VARCHAR(50) NOT NULL
    -> );
Query OK, 0 rows affected (0.027 sec)

MariaDB [sd]> CREATE TABLE Employee (
```

```
    ->     EmpID INT PRIMARY KEY,
    ->     EmpName VARCHAR(100) NOT NULL,
    ->     Salary DECIMAL(10,2),
    ->     DeptID INT,
    ->     FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
    -> );
Query OK, 0 rows affected (0.061 sec)

MariaDB [sd]> -- Insert into Department table
MariaDB [sd]> INSERT INTO Department (DeptID, DeptName) VALUES (1, 'HR');
Query OK, 1 row affected (0.011 sec)

MariaDB [sd]> INSERT INTO Department (DeptID, DeptName) VALUES (2, 'IT');
Query OK, 1 row affected (0.003 sec)

MariaDB [sd]> INSERT INTO Department (DeptID, DeptName) VALUES (3, 'Finance');
Query OK, 1 row affected (0.003 sec)

MariaDB [sd]>
MariaDB [sd]> -- Insert into Employee table
MariaDB [sd]> INSERT INTO Employee (EmpID, EmpName, Salary, DeptID) VALUES (101,
'Anushika', 30000, 2);
Query OK, 1 row affected (0.010 sec)

MariaDB [sd]> INSERT INTO Employee (EmpID, EmpName, Salary, DeptID) VALUES (102,
'Ravi', 25000, 1);
Query OK, 1 row affected (0.004 sec)

MariaDB [sd]> INSERT INTO Employee (EmpID, EmpName, Salary, DeptID) VALUES (103,
'Divya', 40000, 3);
Query OK, 1 row affected (0.005 sec)
```

Output:

```
MariaDB [sd]> SELECT * FROM Department;
+--------+----------+
| DeptID | DeptName |
+--------+----------+
|      1 | HR       |
|      2 | IT       |
|      3 | Finance  |
+--------+----------+
3 rows in set (0.007 sec)

MariaDB [sd]>
```

```
MariaDB [sd]> SELECT * FROM Employee;
+-------+----------+----------+--------+
| EmpID | EmpName  | Salary   | DeptID |
+-------+----------+----------+--------+
|   101 | Anushika | 30000.00 |      2 |
|   102 | Ravi     | 25000.00 |      1 |
|   103 | Divya    | 40000.00 |      3 |
+-------+----------+----------+--------+
3 rows in set (0.001 sec)

MariaDB [sd]>
```

5)DAY-10

Write queries to search and sort Employee data

```
MariaDB [sd]> SELECT * FROM Employee
    -> WHERE EmpName = 'Anushika';
+-------+----------+----------+--------+
| EmpID | EmpName  | Salary   | DeptID |
+-------+----------+----------+--------+
|   101 | Anushika | 30000.00 |      2 |
+-------+----------+----------+--------+
1 row in set (0.001 sec)

MariaDB [sd]> SELECT * FROM Employee
    -> WHERE DeptID = 2;
+-------+----------+----------+--------+
| EmpID | EmpName  | Salary   | DeptID |
+-------+----------+----------+--------+
|   101 | Anushika | 30000.00 |      2 |
+-------+----------+----------+--------+
1 row in set (0.005 sec)

MariaDB [sd]> SELECT * FROM Employee
    -> WHERE Salary BETWEEN 25000 AND 40000;
+-------+----------+----------+--------+
| EmpID | EmpName  | Salary   | DeptID |
+-------+----------+----------+--------+
|   101 | Anushika | 30000.00 |      2 |
|   102 | Ravi     | 25000.00 |      1 |
|   103 | Divya    | 40000.00 |      3 |
+-------+----------+----------+--------+
3 rows in set (0.001 sec)

MariaDB [sd]> SELECT * FROM Employee
    -> ORDER BY Salary ASC;
```

```
+-------+----------+----------+--------+
| EmpID | EmpName  | Salary   | DeptID |
+-------+----------+----------+--------+
|   102 | Ravi     | 25000.00 |      1 |
|   101 | Anushika | 30000.00 |      2 |
|   103 | Divya    | 40000.00 |      3 |
+-------+----------+----------+--------+
3 rows in set (0.004 sec)

MariaDB [sd]> SELECT * FROM Employee
    -> ORDER BY EmpName ASC;
+-------+----------+----------+--------+
| EmpID | EmpName  | Salary   | DeptID |
+-------+----------+----------+--------+
|   101 | Anushika | 30000.00 |      2 |
|   103 | Divya    | 40000.00 |      3 |
|   102 | Ravi     | 25000.00 |      1 |
+-------+----------+----------+--------+
3 rows in set (0.001 sec)

MariaDB [sd]> SELECT * FROM Employee
    -> ORDER BY DeptID ASC, Salary DESC;
+-------+----------+----------+--------+
| EmpID | EmpName  | Salary   | DeptID |
+-------+----------+----------+--------+
|   102 | Ravi     | 25000.00 |      1 |
|   101 | Anushika | 30000.00 |      2 |
|   103 | Divya    | 40000.00 |      3 |
+-------+----------+----------+--------+
3 rows in set (0.001 sec)

MariaDB [sd]> SELECT * FROM Employee
    -> WHERE EmpName LIKE 'A%';
+-------+----------+----------+--------+
| EmpID | EmpName  | Salary   | DeptID |
+-------+----------+----------+--------+
|   101 | Anushika | 30000.00 |      2 |
+-------+----------+----------+--------+
1 row in set (0.011 sec)

MariaDB [sd]>
```