

Assignment 5

1. Design and implement a class named `InstanceCounter` to track and count the number of instances created from this class.

Code :-

```
package project;

public class InstanceCounter {

    //static variable
    private static int instanceCount;

    static {
        instanceCount = 0;
        System.out.println("Static initializer: Instance count initialized.");
    }

    // Non-static variable
    private int id;

    // Constructor increments the instance count when a new object is created
    public InstanceCounter() {
        instanceCount++;
        this.id = instanceCount;
        System.out.println("Constructor: Created instance #" + id);
    }

    // Static method to return the number of instances created
    public static int getInstanceCount() {
        return instanceCount;
    }

    // Getter id
    public int getId() {
        return id;
    }

    // Setter id
    public void setId(int id) {
        this.id = id;
    }

    // toString()
    public String toString() {
        return "InstanceCounter{id=" + id + "}";
    }

    // Main method
    public static void main(String[] args) {
        // Create new instances
    }
```

```

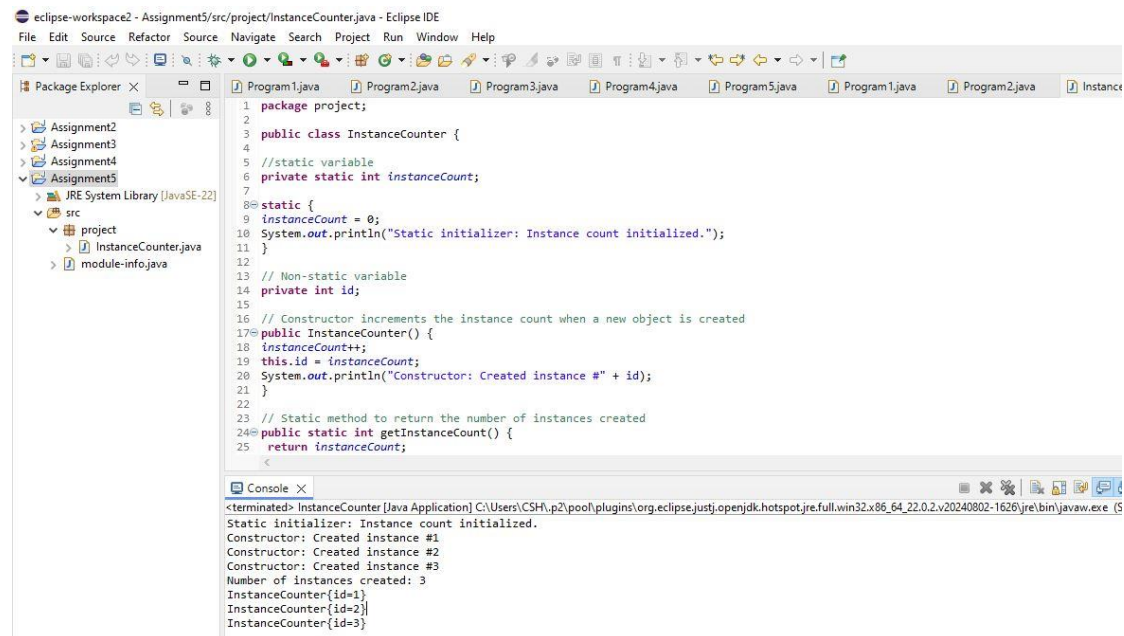
InstanceCounter obj1 = new InstanceCounter();
InstanceCounter obj2 = new InstanceCounter();
InstanceCounter obj3 = new InstanceCounter();

// Output of instances created
System.out.println("Number of instances created: " +
InstanceCounter.getInstanceCount());

// Display
System.out.println(obj1);
System.out.println(obj2);
System.out.println(obj3);
    }
}

```

Output –



The screenshot shows the Eclipse IDE with the 'InstanceCounter.java' file open. The code in the editor is as follows:

```

1 package project;
2
3 public class InstanceCounter {
4
5     //static variable
6     private static int instanceCount;
7
8     static {
9         instanceCount = 0;
10        System.out.println("Static initializer: Instance count initialized.");
11    }
12
13    // Non-static variable
14    private int id;
15
16    // Constructor increments the instance count when a new object is created
17    public InstanceCounter() {
18        instanceCount++;
19        this.id = instanceCount;
20        System.out.println("Constructor: Created instance #" + id);
21    }
22
23    // Static method to return the number of instances created
24    public static int getInstanceCount() {
25        return instanceCount;
26    }
27 }

```

The console output at the bottom shows the following sequence of events:

```

<terminated> InstanceCounter [Java Application] C:\Users\CSH\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_22.0.2.v20240802-1626\jre\bin\javaw.exe (S
Static initializer: Instance count initialized.
Constructor: Created instance #1
Constructor: Created instance #2
Constructor: Created instance #3
Number of instances created: 3
InstanceCounter{id=1}
InstanceCounter{id=2}
InstanceCounter{id=3}

```

- Design and implement a class named `Employee` to manage employee data for a company. The class should include fields to keep track of the total number of employees and the total salary expense, as well as individual employee details such as their ID, name, and salary.

The class should have methods to:

- Retrieve the total number of employees (`getTotalEmployees()`)
- Apply a percentage raise to the salary of all employees (`applyRaise(double percentage)`)
- Calculate the total salary expense, including any raises (`calculateTotalSalaryExpense()`)
- Update the salary of an individual employee (`updateSalary(double newSalary)`)

Understand the problem statement and use static and non-static fields and methods appropriately. Implement static and non-static initializers, constructors, getter and setter methods, and a `toString()` method to handle the initialization and representation of employee data.

Write a menu-driven program in the `main` method to test the functionalities.

Code :-

```
package project;
import java.util.Scanner;
public class Employee {

    // Static fields
    private static int totalEmployees = 0;
    private static double totalSalaryExpense = 0.0;

    // Non-static fields
    private int employeeId;
    private String name;
    private double salary;

    static {
        System.out.println("Employee management system started...");
    }

    // Constructor
    public Employee(int employeeId, String name, double salary) {
        this.employeeId = employeeId;
        this.name = name;
        this.salary = salary;
        totalEmployees++;
        totalSalaryExpense += salary;
    }

    // Getters and Setters
    public int getEmployeeId() {
        return employeeId;
    }

    public void setEmployeeId(int employeeId) {
        this.employeeId = employeeId;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public double getSalary() {
        return salary;
    }
}
```

```

public void setSalary(double salary) {

    totalSalaryExpense -= this.salary; // Subtract current salary
    this.salary = salary;
    totalSalaryExpense += salary; // Add new salary
}

// Static method
public static int getTotalEmployees() {
    return totalEmployees;
}

public static void applyRaise(Employee[] employees, double percentage) {
    for (Employee emp : employees) {
        double newSalary = emp.salary + (emp.salary * percentage / 100);
        emp.setSalary(newSalary); // Update the employee's salary
    }
}

// to calculate the total salary
public static double calculateTotalSalaryExpense() {
    return totalSalaryExpense;
}

    public String toString() {
    return "Employee ID: " + employeeId + ", Name: " + name + ", Salary: $" + salary;
    }

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

    Employee[] employees = new Employee[3];

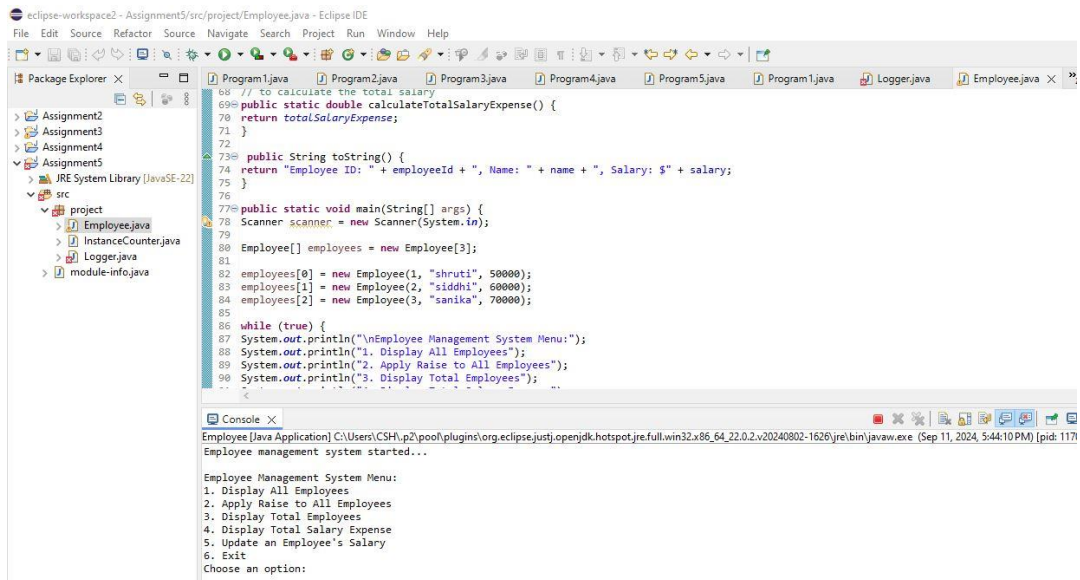
    employees[0] = new Employee(1, "shruti", 50000);
    employees[1] = new Employee(2, "siddhi", 60000);
    employees[2] = new Employee(3, "sanika", 70000);

    while (true) {
        System.out.println("\nEmployee Management System Menu:");
        System.out.println("1. Display All Employees");
        System.out.println("2. Apply Raise to All Employees");
        System.out.println("3. Display Total Employees");
        System.out.println("4. Display Total Salary Expense");
        System.out.println("5. Update an Employee's Salary");
        System.out.println("6. Exit");
        System.out.print("Choose an option: ");
        int choice = scanner.nextInt();

        }
    }
}

```

Output



The screenshot displays the Eclipse IDE interface. The Package Explorer on the left shows a project named 'project' containing several Java files, including 'Employee.java'. The main editor window shows the code for 'Employee.java', which includes methods for calculating total salary expense, converting an employee to a string, and a main method that initializes an array of employees and displays a menu. The Console window at the bottom shows the output of the program, indicating that the employee management system has started and displaying the menu options.

```
eclipse-workspace2 - Assignment5/src/project/Employee.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help

Package Explorer
> Assignment2
> Assignment3
> Assignment4
> Assignment5
  JRE System Library [JavaSE-22]
  src
    project
      Employee.java
      InstanceCounter.java
      Logger.java
      module-info.java

Program1.java Program2.java Program3.java Program4.java Program5.java Program1.java Logger.java Employee.java
68 // to calculate the total salary
69 public static double calculateTotalSalaryExpense() {
70     return totalSalaryExpense;
71 }
72
73 public String toString() {
74     return "Employee ID: " + employeeId + ", Name: " + name + ", Salary: $" + salary;
75 }
76
77 public static void main(String[] args) {
78     Scanner scanner = new Scanner(System.in);
79
80     Employee[] employees = new Employee[3];
81
82     employees[0] = new Employee(1, "shruti", 50000);
83     employees[1] = new Employee(2, "siddhi", 60000);
84     employees[2] = new Employee(3, "sanika", 70000);
85
86     while (true) {
87         System.out.println("\nEmployee Management System Menu:");
88         System.out.println("1. Display All Employees");
89         System.out.println("2. Apply Raise to All Employees");
90         System.out.println("3. Display Total Employees");
91     }
92 }

Employee [Java Application] C:\Users\CSH\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_22.0.2.v20240802-1626\jre\bin\javaw.exe (Sep 11, 2024, 5:44:10 PM) [pid: 117]
Employee management system started...

Employee Management System Menu:
1. Display All Employees
2. Apply Raise to All Employees
3. Display Total Employees
4. Display Total Salary Expense
5. Update an Employee's Salary
6. Exit
Choose an option:
```