}

ASSIGNMENT NO.5

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

Ans.
```
package org.example.question1;
class InstanceCounter {
   // Static field to keep track of the number of instances
   private static int instanceCount = 0;

   // Constructor
   public InstanceCounter() {
      // Increment the instance count whenever a new instance is created
      instanceCount++;
   }

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

   // Overriding toString() to provide information about the instance
   @Override
   public String toString() {
      return "InstanceCounter instance created. Total instances: " + instanceCount;
   }
}
public class Program1 {

         public static void main(String[] args) {
                   InstanceCounter obj1 = new InstanceCounter();
            InstanceCounter obj2 = new InstanceCounter();
            InstanceCounter obj3 = new InstanceCounter();


               System.out.println("Current instance count: " +
InstanceCounter.getInstanceCount());


               System.out.println(obj1);
               System.out.println(obj2);
               System.out.println(obj3);

         }

}
```

2. Design and implement a class named `Logger` to manage logging messages for an application. The class should be implemented as a singleton to ensure that only one instance of the `Logger` exists throughout the application.

   The class should include the following methods:

   - **`getInstance()`**: Returns the unique instance of the `Logger` class.
   - **`log(String message)`**: Adds a log message to the logger.
   - **`getLog()`**: Returns the current log messages as a `String`.
   - **`clearLog()`**: Clears all log messages.

Ans. package org.example.question2;

```
class Logger{

    private static Logger instance;



    private StringBuilder logMessages;



    private Logger() {

        logMessages = new StringBuilder();



    }



    static {

        instance = new Logger();

    }



    public static Logger getInstance() {

        return instance;

    }
```

```
            public void log(String message) {
                logMessages.append(message).append("\n");
            }
            public String getLog() {
                return logMessages.toString();
            }

            public void clearLog() {
                logMessages.setLength(0);
            }
}



public class Program2 {

            public static void main(String[] args) {
                    Logger logger = Logger.getInstance();

                    logger.log("application started.");
                    logger.log("performing operations.");
                    logger.log("operations successfull.");

                    System.out.println("Current log :  ");
```

```
        System.out.println(logger.getLog());



        logger.clearLog();



        System.out.println("log after clearing: ");

        System.out.println(logger.getLog());

    }


}
```

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.

   Ans. package org.example.question3;

```
//import java.security.PublicKey;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;


class Employee{

    private static int totalEmployee = 0;

    private static double totalSalaryExpense = 0;

    private static List<Employee> employeeList = new ArrayList<>();

    private int id;

    private String name;

    private double salary;


    static {

      System.out.println("Employee management system initialized.");

  }


    public Employee(int id, String name, double salary) {


            this.id = id;

            this.name = name;

            this.salary = salary;

            totalEmployee++;

            totalSalaryExpense = totalSalaryExpense+salary;
            employeeList.add(this);



    }

    public static int getTotalEmployees() {
            return totalEmployee;
    }

     public static void applyRaise(double percentage) {
          for (Employee emp : employeeList) {
             double raiseAmount = emp.salary * (percentage / 100);
             emp.salary += raiseAmount;
          }
          calculateTotalSalaryExpense(); // Recalculate total salary expense
       }

    static double calculateTotalSalaryExpense() {
            totalSalaryExpense = 0;
      for (Employee emp : employeeList) {
         totalSalaryExpense += emp.salary;
      }
      return totalSalaryExpense;

    }
```

```
  public void updateSalary(double newSalary) {
    totalSalaryExpense -= this.salary; // Remove old salary from total
    this.salary = newSalary;
    totalSalaryExpense += this.salary; // Add new salary to total
}

  public int getId() {
        return id;
  }

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

  public String getName() {
        return name;
  }

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

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

```
	public void setSalary(double salary) {
		this.salary = salary;
	}

	@Override
	public String toString() {
		return "Employee [ID =" +id + ", Name = " +name +", Salary = "+salary+" ]";
	}

	}



public class Program3 {

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

		while(running) {
			System.out.println("choose from the following :");
			System.out.println("1. Add Employee :");
			System.out.println("2. Apply Raise :");
System.out.println("3. Update employee Salary :");
System.out.println("4. Display total Employee :");
System.out.println("5. Display total salary Expense :");
System.out.println("6. Display all Employees");
System.out.println("7. Exit :");

int choice = sc.nextInt();
sc.nextLine();



switch(choice) {
case 1 :
        System.out.println("Enter employee ID: ");
        int id = sc.nextInt();
        System.out.println("Enter employee name: ");
        String name = sc.nextLine();
        sc.nextLine();
        System.out.println("Enter employee Salary :");
        double salary = sc.nextDouble();
        new Employee(id , name , salary);
        System.out.println("Employee added successfully.");
        break;
 case 2:
System.out.print("Enter raise percentage: ");
double percentage = sc.nextDouble();
```

```
    Employee.applyRaise(percentage);

    System.out.println("Raise applied to all employees.");

    break;


case 3:

    System.out.print("Enter employee ID to update salary: ");

    int empId = sc.nextInt();

    sc.nextLine(); // Consume newline

    boolean found = false;

    for (Employee emp : employeeList) {

        if (emp.getId() == empId) {

            System.out.print("Enter new salary: ");

            double newSalary = sc.nextDouble();

            emp.updateSalary(newSalary);

            System.out.println("Salary updated.");

            found = true;

            break;

        }

    }

    if (!found) {

        System.out.println("Employee not found.");

    }

    break;


case 4:
```

```
                System.out.println("Total number of employees: " +
Employee.getTotalEmployees());

                break;


            case 5:

                System.out.println("Total salary expense: " +
Employee.calculateTotalSalaryExpense());

                break;


            case 6:

                System.out.println("Employee List:");

                for (Employee emp : employeeList) {

                    System.out.println(emp);

                }

                break;


            case 7:

                running = false;

                System.out.println("Exiting program.");

                break;


            default:

                System.out.println("Invalid choice. Please try again.");

                        }

                 }

                 sc.close();
```

```
        }


    }
```