ASSIGNMENT NO.6

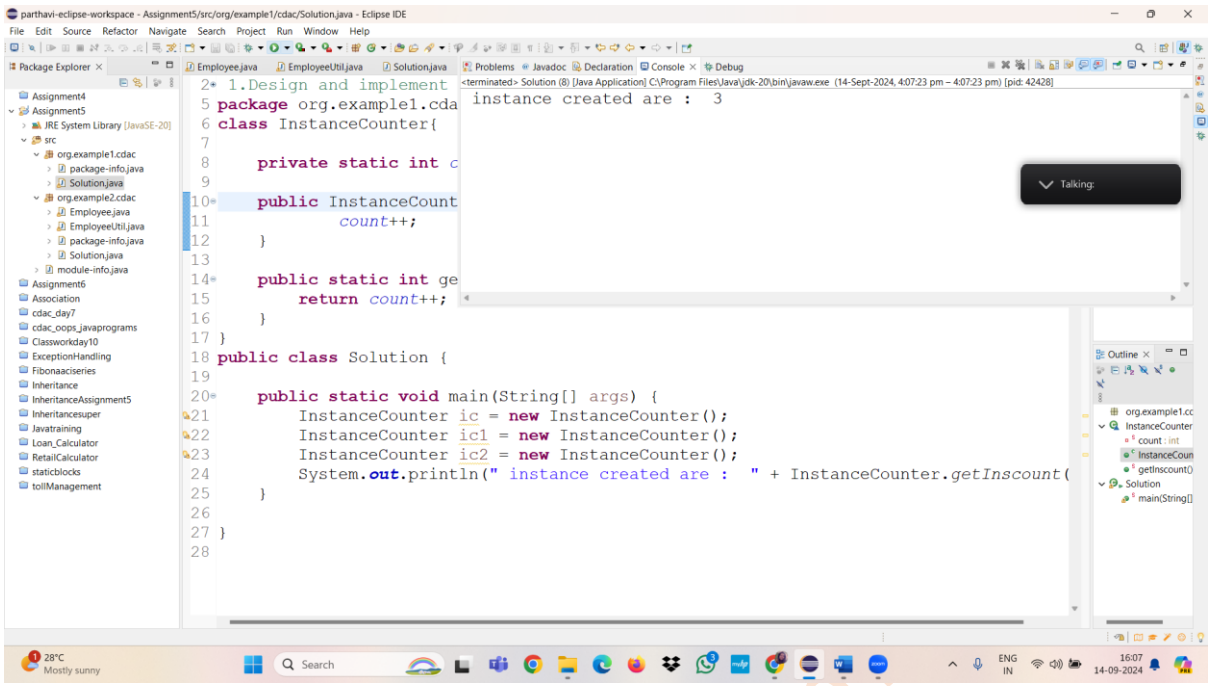
Note:

1. This assignment is designed to practice static fields, static initializers, and static methods.
2. Understand the problem statement and use static and non-static wisely to solve the problem.
3. Use constructors, proper getter/setter methods, and `toString()` wherever required.

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

```
package org.example1.cdac;
class InstanceCounter{

	private static int count =0;

	public InstanceCounter(){
			count++;
	}

	public static int getInscount() {
		return count++;
	}
}
public class Solution {

	public static void main(String[] args) {
		InstanceCounter ic = new InstanceCounter();
		InstanceCounter ic1 = new InstanceCounter();
		InstanceCounter ic2 = new InstanceCounter();
		System.out.println(" instance created are :  " + InstanceCounter.getInscount());
	}

}
```

ASSIGNMENT NO.6



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.

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)`)

ASSIGNMENT NO.6

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.

```
package org.example2.cdac;
/*
 public void setSalary(double salary) {
     // Adjust total salary expense to reflect the new salary
     totalSalaryExpense = totalSalaryExpense - this.salary + salary;
     this.salary = salary;
  }
   public static void applyRaise(double percentage) {
     // Multiply the total salary expense by the raise percentage
     double raiseAmount = totalSalaryExpense * (percentage / 100);
     totalSalaryExpense += raiseAmount;
  }

 */
public class Employee {

        String name;
        int empid;
        double salary;
        static double totalsal;
        static int numemp;
        static double percentage;

        public Employee() {
                this(" ", 0, 0.0, 0.0);
        }

        static{

                numemp = 0;
                percentage = 0.0;

        }
        public Employee(String name, int empid, double salary, double totalsal ) {
                this.name = name;
                this.empid = empid;
                this.salary = salary;
                numemp++;
                Employee.totalsal = totalsal;
```

```
	}

	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public int getEmpid() {
		return empid;
	}
	public void setEmpid(int empid) {
		this.empid = empid;
	}
	public double getSalary() {
		return salary;
	}
	public void setSalary(double salary) {

		totalsal = totalsal - this.salary + salary;
		this.salary = salary;
	}
	public double getTotalsal() {
		return totalsal;
	}

	 public static void applyRaise(double percentage) {
		  double raiseAmount = totalsal * (percentage / 100);
		totalsal += raiseAmount;
	   }

//	public int getNumemp() {
//		return this.numemp;
//	}

	public double getPercentage() {
		return percentage;
	}



	@Override
	public String toString() {
		return this.name + " " + this.empid + " " + this.salary + "  " + this.totalsal;

	}
}
```

```
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
package org.example2.cdac;

import java.util.Scanner;

public class EmployeeUtil {

	private Employee em = new Employee();
	Scanner sc = new Scanner(System.in);
	public void acceptRec() {
		//Scanner sc = new Scanner(System.in);
		System.out.print(" enter name  :  ");
		this.em.setName(sc.nextLine());
		sc.nextLine();
		System.out.println(" enter id : ");
		this.em.setEmpid(sc.nextInt());
		System.out.println(" enter salary :  ");
		this.em.setSalary(sc.nextDouble());
		System.out.println(" enter percentage raise in salary  : ");
		Employee.percentage = sc.nextDouble();
		Employee.applyRaise(Employee.percentage);

	}

	public void printrec() {
		System.out.print(" Name  :  " + this.em.name);
		System.out.println("Id : " + this.em.empid);
		System.out.println(" Salary :  " + this.em.salary);
		System.out.println(" Total salary :  " + this.em.totalsal);
		System.out.println(" Raise in salary  : " + this.em.getTotalsal());
		System.out.println(" Final salary is :  " + Employee.totalsal);

	}

	public int menulist() {

		System.out.println(" 0  : EXIT ");
		System.out.println(" 1  : Accept record ");
		System.out.println(" 2  : Print record ");
		System.out.print(" 3  : Enter choice ");
		int choice = sc.nextInt();
		return choice;
	}
}
```

ASSIGNMENT NO.6

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

```
public class Solution {

	public static void main(String[] args) {
		EmployeeUtil ut = new EmployeeUtil();
		int choice;
		while ((choice = ut.menulist()) != 0) {
			switch(choice) {
			case 1:
				ut.acceptRec();
			break;

			case 2:
				ut.printrec();
			break;
			}

		}
	}

}
```