**CDAC MUMBAI**

**MODULE 2 OOPJ**

**ASSIGNMENT 5**

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

Program:

**package** org.Assignment5;

**class** InstanceCounter {

**private** **static** **int** *instanceCounterValue*=0;

**public** InstanceCounter() {

*instanceCounterValue*++;

}

**public** **static** **int** getInstanceCounterValue() {

**return** *instanceCounterValue*;

}

}

**public** **class** InstanceCounterProgram{

**public** **static** **void** main(String[] args) {

InstanceCounter IC1=**new** InstanceCounter();

InstanceCounter IC2=**new** InstanceCounter();

InstanceCounter IC3=**new** InstanceCounter();

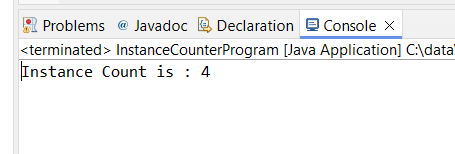
InstanceCounter IC4=**new** InstanceCounter();

System.***out***.println("Instance Count is : "+ InstanceCounter.*getInstanceCounterValue*() );

}

}

Output:



1. 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.

**package** org.Assignment5;

**public** **class** Logger {

**private** **static** Logger *instance*;

**private** StringBuilder logMessages;

**private** Logger() {

logMessages = **new** StringBuilder();

}

**public** **static** Logger getInstance() {

**if** (*instance* == **null**) {

**synchronized** (Logger.**class**) {

**if** (*instance* == **null**) {

*instance* = **new** Logger();

}

}

}

**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** **static** **void** main(String[] args) {

Logger logger = Logger.*getInstance*();

logger.log("First log message");

logger.log("Second log message");

System.***out***.println("Current log messages:");

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

logger.clearLog();

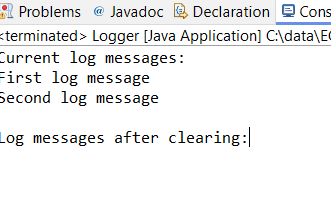
System.***out***.println("Log messages after clearing:");

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

}

}

Output:



1. 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.

package org.Assignment5;

import java.util.ArrayList;

import java.util.List;

public class Employee {

private static int totalEmployees = 0;

private static double totalSalaryExpense = 0.0;

private int id;

private String name;

private double salary;

static {

totalEmployees = 0;

totalSalaryExpense = 0.0;

}

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

this.id = id;

this.name = name;

this.salary = salary;

totalEmployees++;

totalSalaryExpense += salary;

}

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) {

totalSalaryExpense = totalSalaryExpense - this.salary;

this.salary = salary;

totalSalaryExpense += salary;

}

public static int getTotalEmployees() {

return totalEmployees;

}

public static void applyRaise(List<Employee> employees, double percentage) {

for (Employee employee : employees) {

double newSalary = employee.getSalary() \* (1 + percentage / 100);

employee.setSalary(newSalary);

}

}

public static double calculateTotalSalaryExpense() {

return totalSalaryExpense;

}

@Override

public String toString() {

return String.format("Employee [ID=%d Name=%s Salary=%.2f ]", id, name, salary);

}

public static void main(String[] args) {

List<Employee> employees = new ArrayList<>();

employees.add(new Employee(1, "ABC XYZ", 50000));

employees.add(new Employee(2, "PQR WXY", 60000));

employees.add(new Employee(3, "LMN OPQ", 55000));

System.out.println("Total Employees: " + Employee.getTotalEmployees());

System.out.println("Total Salary Expense: " + Employee.calculateTotalSalaryExpense());

System.out.println("\nApplying a 10% raise to all employees...");

Employee.applyRaise(employees, 10);

System.out.println("Total Salary Expense after raise: " + Employee.calculateTotalSalaryExpense());

System.out.println("\nEmployee Details:");

for (Employee employee : employees) {

System.out.println(employee);

}

}

}

Output:

