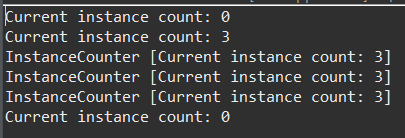
**Note:**

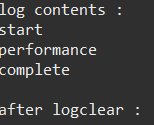
1. This assignment is designed to practice static fields, static initializers, and static methods.
2. Understand the problem statement and use static and non-static wisely to solve the problem.
3. Use constructors, proper getter/setter methods, and toString() wherever required.
4. Design and implement a class named InstanceCounter to track and count the number of instances created from this class.
5. package org.question1;
6. class InstanceCounter {
7. private static int *instanceCount* = 0;
8. public InstanceCounter() {
9. *instanceCount*++;
10. }
11. public static int getInstanceCount() {
12. return *instanceCount*;
13. }
14. public static void resetInstanceCount() {
15. *instanceCount* = 0;
16. }
17. public static void displayInstanceCount() {
18. System.***out***.println("Current instance count: " + *instanceCount*);
19. }
20. public String toString() {
21. return String.*format*("InstanceCounter [Current instance count: %d]", *instanceCount*);
22. }
23. }
24. public class InstanceCounterTester {
25. public static void main(String[] args) {
27. InstanceCounter.*displayInstanceCount*();
28. InstanceCounter obj1 = new InstanceCounter();
29. InstanceCounter obj2 = new InstanceCounter();
30. InstanceCounter obj3 = new InstanceCounter();
31. InstanceCounter.*displayInstanceCount*();
32. System.***out***.println(obj1);
33. System.***out***.println(obj2);
34. System.***out***.println(obj3);
35. InstanceCounter.*resetInstanceCount*();
36. InstanceCounter.*displayInstanceCount*();
37. }
38. }



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.question2;
* public class Logger {
* private static final Logger ***instance*** = new Logger();
* private String message;
* private Logger() {
* message = "";
* }
* public static Logger getInstance() {
* return ***instance***;
* }
* public void logMessage(String msg) {
* message += msg + System.*lineSeparator*();
* }
* public String getLog() {
* return message;
* }
* public void clearLog() {
* message = "";
* }
* }
* package org.question2;
* public class Program {
* public static void main(String[] args) {
* Logger logger = Logger.*getInstance*();
* logger.logMessage("\nstart");
* logger.logMessage("performance");
* logger.logMessage("complete");
* System.***out***.println("log contents : "+ logger.getLog());
* logger.clearLog();
* System.***out***.println("after logclear : "+ logger.getLog());
* // Logger log2 = Logger.getInstance();
* // log2.logMessage("\nstart");
* // log2.logMessage("performance");
* // log2.logMessage("complete");
* //
* // System.out.println("log contents : "+ log2.getLog());
* //
* // log2.clearLog();
* // System.out.println("after logclear : "+ log2.getLog());
* }
* }



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.question3;

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 static int getTotalEmployees() {

return *totalEmployees*;

}

public static double calculateTotalSalaryExpense() {

return *totalSalaryExpense*;

}

public static void applyRaise(double percentage) {

double raiseAmount = *totalSalaryExpense* \* (percentage/100);

*totalSalaryExpense* += raiseAmount;

}

public void UpdateSalary(double newSalary) {

*totalSalaryExpense* = *totalSalaryExpense* - this.salary + newSalary;

this.salary = newSalary;

}

public String toString() {

return "Employee [id=" + this.id + ", name=" + this.name + ", salary=" + this.salary + "]";

}

}

package org.question3;

import java.util.Scanner;

public class Program {

public static void main(String[] args) {

Scanner sc = new Scanner(System.***in***);

while(true) {

System.***out***.println("Menu:");

System.***out***.println("1. add employee:");

System.***out***.println("2. apply raise to all employees:");

System.***out***.println("3. update salary of an employee:");

System.***out***.println("4. show total number of employees:");

System.***out***.println("5. show total salary expense:");

System.***out***.println("6. show employee data:");

System.***out***.println("7. exit:");

System.***out***.println("choose an option:");

int choice = sc.nextInt();

sc.nextLine();

switch(choice){

case 1:

System.***out***.print("Enter employee ID: ");

int id = sc.nextInt();

sc.nextLine();

System.***out***.print("Enter employee name: ");

String name = sc.nextLine();

System.***out***.print("Enter employee salary: ₹");

double salary = sc.nextDouble();

sc.nextLine();

new Employee(id, name, salary);

System.***out***.println("Employee added.");

break;

case 2:

System.***out***.print("Enter percentage raise: ");

double percentage = sc.nextDouble();

sc.nextLine();

Employee.*applyRaise*(percentage);

System.***out***.println("Raise applied.");

break;

case 3:

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

id = sc.nextInt();

sc.nextLine();

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

salary = sc.nextDouble();

sc.nextLine();

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

break;

case 4:

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

break;

case 5:

System.***out***.printf("Total salary expense: ₹%.2f%n", Employee.*calculateTotalSalaryExpense*());

break;

// case 6:

// System.out.println(Employee.toString());

// break;

case 6:

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

sc.close();

return;

default:

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

break;

}

}

}

}

