**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.

**package** org.example;

**public** **class** InstanceCounter {

**private** **static** **int** *instanceCount* = 0;

**public** InstanceCounter() {

*instanceCount*++;

}

**public** **static** **int** getInstanceCount() {

**return** *instanceCount*;

}

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

InstanceCounter obj1 = **new** InstanceCounter();

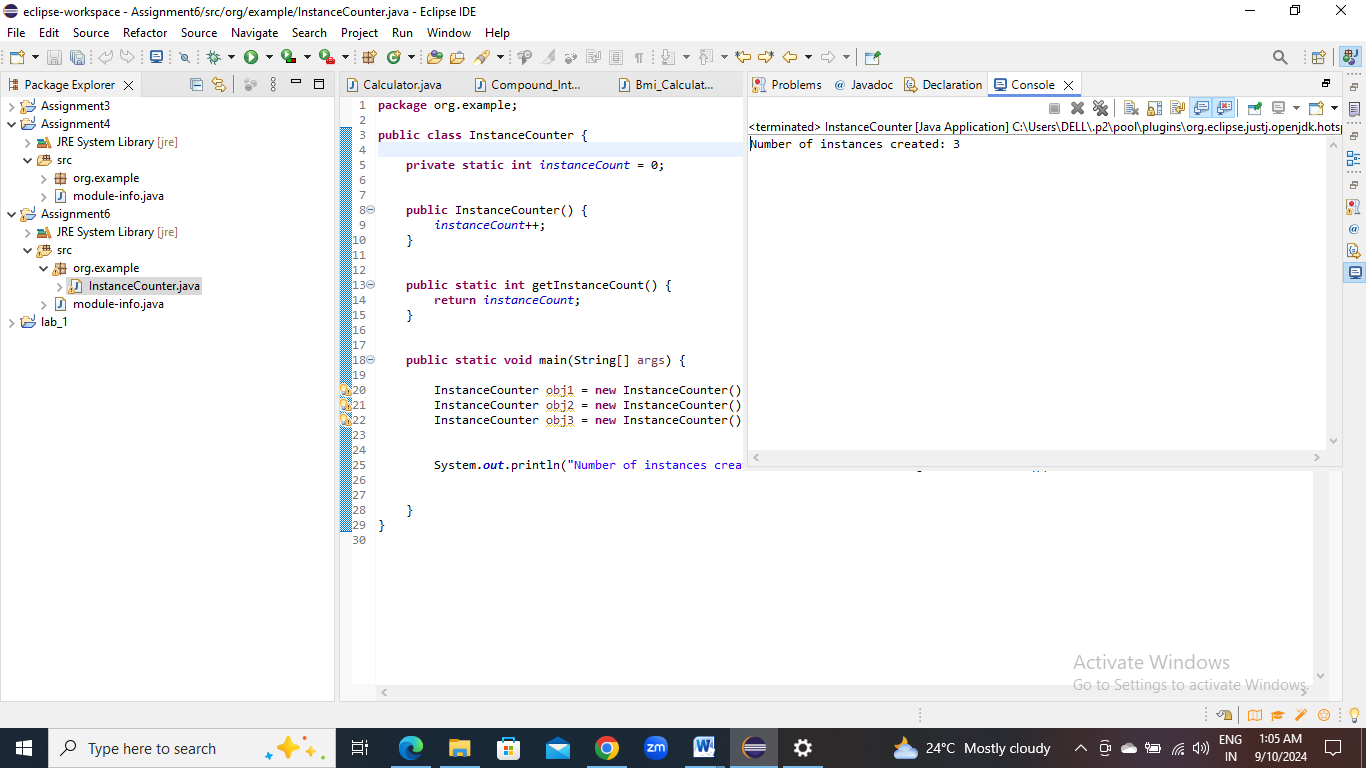
InstanceCounter obj2 = **new** InstanceCounter();

InstanceCounter obj3 = **new** InstanceCounter();

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

}

}



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.example;
* **public** **class** Logger {
* **private** **static** Logger *instance*;
* **private** StringBuilder logMessages;
* **private** Logger() {
* logMessages = **new** StringBuilder();
* }
* **public** **static** **synchronized** Logger getInstance() {
* **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);
* }
* }

**package** org.example;

**public** **class** LoggerTest {

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

Logger logger = Logger.*getInstance*();

logger.log("Application started");

logger.log("User logged in");

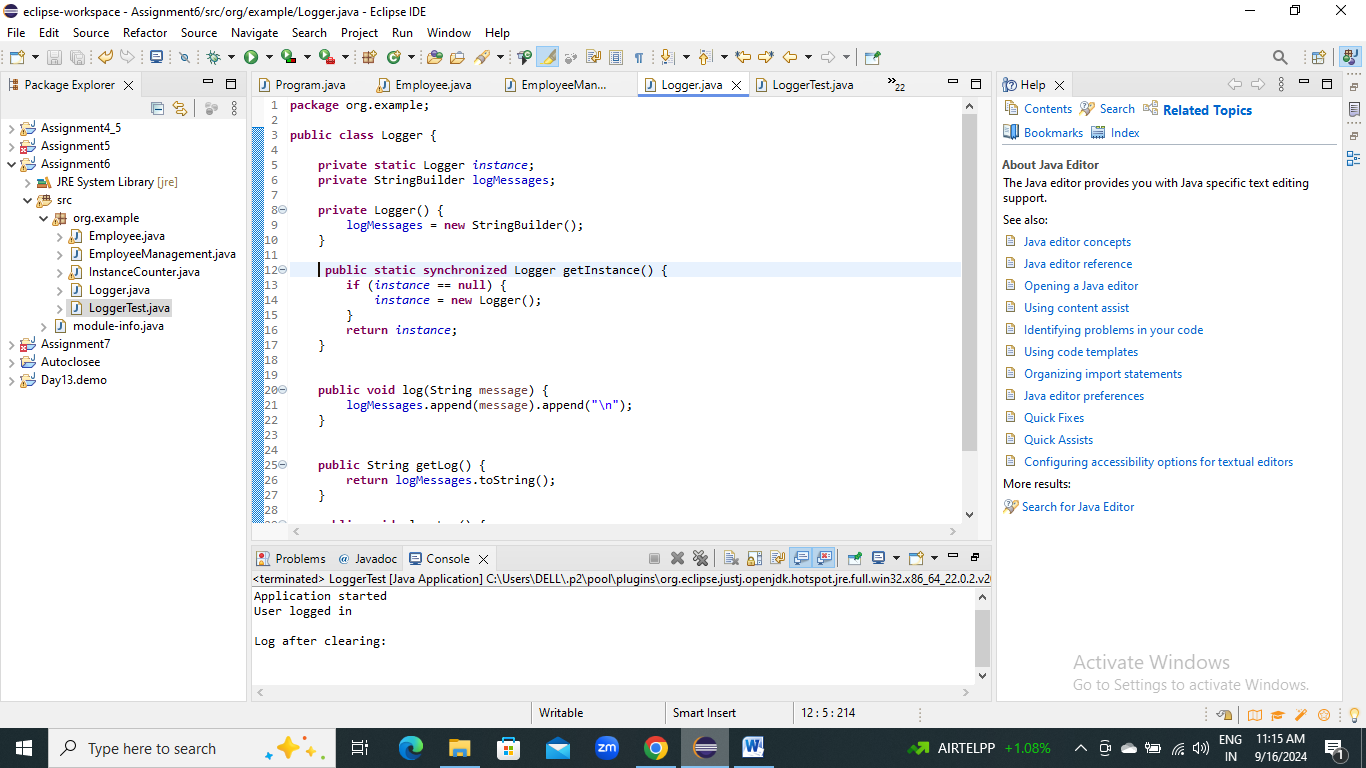
System.***out***.println("Current Log:\n" + logger.getLog());

logger.clearLog();

System.***out***.println("Log after clearing:\n" + logger.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.

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 String getName() {

return name;

}

public double getSalary() {

return salary;

}

public void updateSalary(double newSalary) {

totalSalaryExpense = totalSalaryExpense - this.salary + newSalary;

this.salary = newSalary;

}

public static int getTotalEmployees() {

return totalEmployees;

}

public static double calculateTotalSalaryExpense() {

return totalSalaryExpense;

}

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

for (Employee emp : employees) {

double raiseAmount = emp.salary \* (percentage / 100);

emp.updateSalary(emp.salary + raiseAmount);

}

}

@Override

public String toString() {

return "Employee[ID=" + id + ", Name=" + name + ", Salary=" + salary + "]";

}

}

**package** org.example;

**import** java.util.ArrayList;

**import** java.util.List;

**import** java.util.Scanner;

**public** **class** EmployeeManagement {

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

List<Employee> employeeList = **new** ArrayList<>();

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

**int** choice;

**do** {

System.***out***.println("\nEmployee Management System");

System.***out***.println("1. Add New Employee");

System.***out***.println("2. View Total Employees");

System.***out***.println("3. View Total Salary Expense");

System.***out***.println("4. Update Employee Salary");

System.***out***.println("5. Apply Raise to All Employees");

System.***out***.println("6. Display All Employees");

System.***out***.println("7. Exit");

System.***out***.print("Enter your choice: ");

choice = scanner.nextInt();

**switch** (choice) {

**case** 1:

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

**int** id = scanner.nextInt();

System.***out***.print("Enter Employee Name: ");

scanner.nextLine();

String name = scanner.nextLine();

System.***out***.print("Enter Employee Salary: ");

**double** salary = scanner.nextDouble();

Employee emp = **new** Employee(id, name, salary);

employeeList.add(emp);

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

**break**;

**case** 2:

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

**break**;

**case** 3:

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

**break**;

**case** 4:

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

**int** updateId = scanner.nextInt();

**boolean** found = **false**;

**for** (Employee e : employeeList) {

**if** (e.getId() == updateId) {

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

**double** newSalary = scanner.nextDouble();

e.updateSalary(newSalary);

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

found = **true**;

**break**;

}

}

**if** (!found) {

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

}

**break**;

**case** 5:

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

**double** percentage = scanner.nextDouble();

Employee.*applyRaise*(employeeList, percentage);

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

**break**;

**case** 6:

System.***out***.println("List of Employees:");

**for** (Employee employee : employeeList) {

System.***out***.println(employee);

}

**break**;

**case** 7:

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

**break**;

**default**:

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

}

} **while** (choice != 7);

scanner.close();

}

}

