**Assignment-5**

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.

import java.util.ArrayList;

import java.util.List;

public class Logger {

    private static Logger instance;

    private List<String> logMessages;

    private Logger() {

        logMessages = new ArrayList<>();

    }

    public static Logger getInstance() {

        if (instance == null) {

            instance = new Logger();

        }

        return instance;

    }

    public void log(String message) {

        logMessages.add(message);

    }

    public String getLog() {

        StringBuilder log = new StringBuilder();

        for (String message : logMessages) {

            log.append(message).append("\n");

        }

        return log.toString();

    }

    public void clearLog() {

        logMessages.clear();

    }

    public static void main(String[] args) {

        Logger logger = Logger.getInstance();

        logger.log("Application started.");

        logger.log("User logged in.");

        logger.log("User performed some action.");

        System.out.println("Current Log:");

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

        logger.clearLog();

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

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

    }

}

OUTPUT:-

Current Log:

Application started.

User logged in.

User performed some action.

Log after clearing:

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

public class Main {

public static void main(String[] args) {

InstanceCounter ic1 = new InstanceCounter();

InstanceCounter ic2 = new InstanceCounter();

InstanceCounter ic3 = new InstanceCounter();

System.out.println("Instance count: " + InstanceCounter.getInstanceCount()); // Output: 3

}

}

OUTPUT:-

Instance count: 3

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))**

public class Main {

public static void main(String[] args) {

Employee employee1 = new Employee(1, "John Doe", 50000.0);

Employee.addEmployee(employee1);

Employee employee2 = new Employee(2, "Jane Doe", 60000.0);

Employee.addEmployee(employee2);

Employee employee3 = new Employee(3, "Bob Smith", 70000.0);

Employee.addEmployee(employee3);

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

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

System.out.println("Applying 10% raise...");

Employee.applyRaise(10);

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

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

System.out.println("Updating salary of employee 1 to 55000.0...");

employee1.updateSalary(55000.0);

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

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

System.out.println("Employee 1: " + employee1.toString());

System.out.println("Employee 2: " + employee2.toString());

System.out.println("Employee 3: " + employee3.toString());

}

}

OUTPUT:-

Total employees: 3

Total salary expense: 180000.0

Applying 10% raise...

Total employees: 3

Total salary expense: 198000.0

Updating salary of employee 1 to 55000.0...

Total employees: 3

Total salary expense: 198500.0

Employee 1: Employee{id=1, name='John Doe', salary=55000.0}

Employee 2: Employee{id=2, name='Jane Doe', salary=66000.0}

Employee 3: Employee{id=3, name='Bob Smith', salary=77000.0}

**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.**