Name: Sownthari R P
Date: 08.08.2024

1. Create a Class named Employee program with class variables as companyName, instance variables with employeeName, employeeID , employeeSalary.
2.Use Data Encapsulation and use getters and setters for updating the employeeSalary.
3.Show function overloading to calculate salary of employee with bonus and salary of employee with deduction.
Program:

```
package sample;
public class Employee {
    private static String companyName = "Payoda";

    // Instance variables
    private String employeeName;
    private int employeeID;
    private double employeeSalary;

    // Constructor
    public Employee(String employeeName, int employeeID, double employeeSalary) {
        this.employeeName = employeeName;
        this.employeeID = employeeID;
        this.employeeSalary = employeeSalary;
    }

    // Getter for employeeName
    private String getEmployeeName() {
        return employeeName;
    }
```

```
// Setter for employeeName
private void setEmployeeName(String employeeName) {
    this.employeeName = employeeName;
}

// Getter for employeeID
private int getEmployeeID() {
    return employeeID;
}

// Setter for employeeID
private void setEmployeeID(int employeeID) {
    this.employeeID = employeeID;
}

// Getter for employeeSalary
private double getEmployeeSalary() {
    return employeeSalary;
}

// Setter for employeeSalary
private void setEmployeeSalary(double employeeSalary) {
    this.employeeSalary = employeeSalary;
}

// Getter for companyName
private static String getCompanyName() {
    return companyName;
}
```

```
// Method Overloading to calculate salary with bonus
public double calculateSalary(double bonus) {
    return employeeSalary + bonus;
}

// Method Overloading to calculate salary with deduction
public double calculateSalary(double deduction, boolean isDeduction) {
    if (isDeduction) {
        return employeeSalary - deduction;
    } else {
        return employeeSalary;
    }
}

public static void main(String[] args) {
    // Creating an Employee object
    Employee emp = new Employee("Sownthari", 10453, 50000);

    // Accessing company name
    System.out.println("Company Name: " + Employee.getCompanyName());

    // Accessing employee details
    System.out.println("Employee Name: " + emp.getEmployeeName());
    System.out.println("Employee ID: " + emp.getEmployeeID());
    System.out.println("Employee Salary: " + emp.getEmployeeSalary());

    // Calculating salary with bonus
    double salaryWithBonus = emp.calculateSalary(5000);
    System.out.println("Salary with Bonus: " + salaryWithBonus);

    // Calculating salary with deduction
```

```
        double salaryWithDeduction = emp.calculateSalary(2000, true);
        System.out.println("Salary with Deduction: " + salaryWithDeduction);
    }
}
```

Output:

```
Company Name: Payoda
Employee Name: Sownthari
Employee ID: 10453
Employee Salary: 50000.0
Salary with Bonus: 55000.0
Salary with Deduction: 48000.0
```

4.What are the Microservices – that use this Gateway and Service Discovery methods using the screen shot:

```
spring.application.name=gatewayservice
server.port=8086
eureka.client.service-url.defaultZone=http://localhost:8761/eureka/

spring.cloud.gateway.routes[0].id=user-service
spring.cloud.gateway.routes[0].uri=lb://USER-SERVICE
spring.cloud.gateway.routes[0].predicates[0]=Path=/users/**

spring.cloud.gateway.routes[1].id=order-service
spring.cloud.gateway.routes[1].uri=lb://ORDER-SERVICE
spring.cloud.gateway.routes[1].predicates[0]=Path=/orders/**

spring.cloud.discovery.enabled=true
```

```
spring.application.name=service-registry
server.port=8761
eureka.client.register-with-eureka=false
eureka.client.fetch-registry=false
eureka.instance.hostname=localhost
```

The microservices using this Gateway and Service Discovery setup are:

User Service:

- Identified by the id=user-service.
- The route is defined as uri=lb://USER-SERVICE.
- The path for this service is Path=/users/**.

Order Service:

- Identified by the id=order-service.
- The route is defined as uri=lb://ORDER-SERVICE.
- The path for this service is Path=/orders/**.

The service-registry application is running on port 8761 and serves as the Eureka Server, which manages the service registry.