Actividad de Aprendizaje
Semana 02

Indicaciones Generales:

1. El plagio se sanciona con la suspensión o expulsión del estudiante de la Universidad. Reglamento General de Estudios.
2. Sea cuidadoso con su redacción, la cual formará parte de su calificación (aspectos léxicos, sintácticos, semánticos).
3. Lea bien la pregunta o enunciado antes de proceder a su desarrollo, administre su tiempo eficazmente.

Actividad 1. Identificar y Corregir Infracciones a los Principios SOLID

Objetivo: El objetivo de esta actividad es que los estudiantes analicen un código fuente que infringe los principios SOLID, identifiquen las violaciones y propongan soluciones para corregirlas aplicando los principios de SOLID.

Problema: Un sistema de gestión de empleados en una empresa tiene varias violaciones de los principios SOLID.

Código Fuente:

```
class Empleado {
    private String nombre;
    private String tipo;
    private double salario;

    public Empleado(String nombre, String tipo, double salario) {
        this.nombre = nombre;
        this.tipo = tipo;
        this.salario = salario;
    }

    public double calcularPago() {
        if (tipo.equals("Gerente")) {
            return salario + 1000; // Bono para gerentes
        } else if (tipo.equals("Desarrollador")) {
            return salario;
        } else if (tipo.equals("Practicante")) {
            return salario * 0.5; // Practicantes reciben medio salario
        }
        return salario;
    }

    public void guardarEnBaseDeDatos() {
        System.out.println("Guardando empleado " + nombre + " en la base de datos...");
    }

    public void generarReporte() {
        System.out.println("Generando reporte para el empleado " + nombre + "...");
```

```
    }
}

class SistemaGestionEmpleados {
    public void procesarEmpleado(Empleado empleado) {
        double pago = empleado.calcularPago();
        System.out.println("Pago calculado: " + pago);

        empleado.guardarEnBaseDeDatos();
        empleado.generarReporte();
    }
}

public class Main {
    public static void main(String[] args) {
        Empleado gerente = new Empleado("Juan", "Gerente", 5000);
        Empleado desarrollador = new Empleado("Ana", "Desarrollador", 3000);
        Empleado practicante = new Empleado("Luis", "Practicante", 1000);

        SistemaGestionEmpleados sistema = new SistemaGestionEmpleados();
        sistema.procesarEmpleado(gerente);
        sistema.procesarEmpleado(desarrollador);
        sistema.procesarEmpleado(practicante);
    }
}
```

Instrucciones:

1. Identificar las violaciones de los principios SOLID.

- La clase Empleado viola el principio de responsabilidad única (SRP) porque maneja múltiples responsabilidades: cálculo de pago, almacenamiento en base de datos y generación de reportes.
- El método calcularPago() infringe el principio de abierto/cerrado (OCP) porque si se agrega un nuevo tipo de empleado, se debe modificar la estructura condicional if-else.
- El código rompe el principio de sustitución de Liskov (LSP) porque la clase Empleado depende del atributo tipo para definir su comportamiento, en lugar de utilizar herencia y polimorfismo.
- Se incumple el principio de segregación de interfaces (ISP), ya que Empleado tiene métodos que no todos los empleados necesitan, como guardarEnBaseDeDatos() y generarReporte().
- El principio de inversión de dependencias (DIP) no se respeta, ya que SistemaGestionEmpleados depende directamente de Empleado en lugar de abstraer la persistencia y generación de reportes.

2. Proponer soluciones para corregir las violaciones.

- **Separar responsabilidades**: Crear clases específicas para persistencia y generación de reportes.
- **Eliminar if-else en calcularPago()**: Utilizar herencia y polimorfismo para definir el cálculo de pago en cada tipo de empleado.
- **Usar interfaces para reportes y almacenamiento**: Permitir diferentes implementaciones sin afectar la lógica del sistema.
- **Aplicar inyección de dependencias**: Pasar las dependencias como parámetros en lugar de crearlas dentro de la clase.

3. Implementar las soluciones en el código.
   Se encuentra en mi github
4. Finalmente, subir este ejercicio a su repositorio GitHub con el siguiente nombre POOII-GRUPOX-S2-2, este repositorio debe ser público. Enviar el repositorio de esta primera a actividad.

Fecha límite de presentación: Determinado por el docente.
Integrantes: 1 o 2 estudiantes.