TRABAJO PRÁCTICO 8

Interfaces y Exepciones

Alumno: Ignacio Figueroa – 45.406.120

Tecnicatura Universitaria en Programación – UTN

Materia: Programación II

Comisión: 7

Codigo:

Objetivo

Desarrollar habilidades en el uso de interfaces y manejo de excepciones en Java para fomentar la modularidad, flexibilidad y robustez del código. Comprender la definición e implementación de interfaces como contratos de comportamiento y su aplicación en el diseño orientado a objetos. Aplicar jerarquías de excepciones para controlar y comunicar errores de forma segura. Diferenciar entre excepciones comprobadas y no comprobadas, y utilizar bloques try, catch, finally y throw para garantizar la integridad del programa. Integrar interfaces y manejo de excepciones en el desarrollo de aplicaciones escalables y mantenibles.

PARTE 1 — INTERFACES E-COMMERCE

Pagable.java

```java
package Parte1;

public interface Pagable {

    double calcularTotal();

}
```

Pago.java

```java
package Parte1;

public interface Pago {

    void procesarPago(double monto);

}
```

PagoConDescuento.java

```java
package Parte1;

public interface PagoConDescuento {

    double aplicarDescuento(double monto);

}
```

Notificable.java

```java
package Parte1;

public interface Notificable {

    void notificar(String nombre);

}
```

Producto.java

```java
package Parte1;

public class Producto implements Pagable {

    private String nombre;
    private double precio;

    public Producto(String nombre, double precio) {
        this.nombre = nombre;
        this.precio = precio;
    }

    @Override
    public double calcularTotal() {
        return precio;
    }

    @Override
    public String toString() {
        return "Producto{" + "nombre=" + nombre + ", precio=" + precio + '}';
    }
}
```

Cliente.java

```java
package Parte1;

public class Cliente implements Notificable {

    private String nombre;

    public Cliente(String nombre) {
        this.nombre = nombre;
    }

    @Override
    public void notificar(String mensaje) {
        System.out.println("Notificación para: " + nombre + ": " + mensaje);
    }

}
```

Pedido.java

```java
package Parte1;

import java.util.ArrayList;
import java.util.List;

public class Pedido implements Pagable {

    private List<Producto> productos = new ArrayList<>();
    private Cliente cliente;
    private String estado;

    public Pedido(Cliente cliente, String estado) {
        this.cliente = cliente;
        this.estado = "Pendiente";
    }

    public void agregarProducto(Producto p) {
        productos.add(p);
    }

    @Override
    public double calcularTotal() {
        return productos.stream().mapToDouble(Producto::calcularTotal).sum();
    }

}
```

TarjetaDeCredito.java

```java
package Parte1;

public class TarjetaCredito implements Pago, PagoConDescuento {

    @Override
    public double aplicarDescuento(double monto) {
        return monto * 0.95;
    }

    @Override
    public void procesarPago(double monto) {
        System.out.println("Pago con Tarjeta de Crédito procesado por $" + monto);
    }

}
```

PayPal.java

```java
package Parte1;

public class PayPal implements Pago {

    @Override
    public void procesarPago(double monto) {
        System.out.println("Pago mediante PayPal procesado por $" + monto);
    }
}
```

DemoParte1.java

```java
package Parte1;


public class DemoParte1 {

    public static void run() {
         System.out.println("=== PARTE 1: SISTEMA E-COMMERCE ===");

        Cliente cliente = new Cliente("Ignacio");
        Pedido pedido = new  Pedido(cliente);

        Producto p1 = new Producto("Mouse", 5000);
        Producto p2 = new Producto("Teclado", 12000);

        pedido.agregarProducto(p1);
        pedido.agregarProducto(p2);

        System.out.println("Total pedido: $" + pedido.calcularTotal());

        pedido.cambiarEstado("En preparación");

        PagoConDescuento tarjeta = new TarjetaCredito();
        double conDescuento = tarjeta.aplicarDescuento(pedido.calcularTotal());

        tarjeta.procesarPago(conDescuento);

        PayPal paypal = new PayPal();
        paypal.procesarPago(pedido.calcularTotal());

        System.out.println();
    }

}
```

Salida en consola

```
Output - Tp8-Interfaces-y-Excepciones (run)
run:
=== PARTE 1: SISTEMA E-COMMERCE ===
Total pedido: $17000.0
Notificaci□n para: Ignacio: El pedido cambi□ a estado: En preparaci□n
Pago con Tarjeta de Cr□dito procesado por $16150.0
Pago mediante PayPal procesado por $17000.0

BUILD SUCCESSFUL (total time: 0 seconds)
```

PARTE 2 — EXCEPCIONES

EdadInvalidaException.java

```java
package Parte2;

public class EdadInvalidaException extends Exception {

    public EdadInvalidaException(String mensaje) {
        super(mensaje);
    }

}
```

DemoParte2.java

```java
package Parte2;

import java.io.*;

public class DemoParte2 {

    public static void run() {
        System.out.println("=== PARTE 2: EXCEPCIONES ===");

        // División segura
        try {
            int a = 10;
            int b = 0;
            int resultado = a / b;
            System.out.println("Resultado: " + resultado);
        } catch (ArithmeticException e) {
            System.out.println("Error: División por cero.");
        }

        // Conversión de cadena a número
        try {
            String texto = "abc";
            int numero = Integer.parseInt(texto);
        } catch (NumberFormatException e) {
            System.out.println("Error: El texto no es un número.");
        }

        // Lectura de archivo
        try {
            BufferedReader br = new BufferedReader(new FileReader("archivo.txt"));
        } catch (FileNotFoundException e) {
            System.out.println("Error: El archivo no existe.");
        }

        // Excepción personalizada
        try {
            validarEdad(-5);
        } catch (EdadInvalidaException e) {
            System.out.println("Edad inválida: " + e.getMessage());
        }

        // try-with-resources
        try (BufferedReader br = new BufferedReader(new FileReader("archivo.txt"))) {
            System.out.println("Contenido: " + br.readLine());
        } catch (IOException e) {
            System.out.println("Error leyendo el archivo (try-with-resources).");
        }

        System.out.println();
    }

    private static void validarEdad(int edad) throws EdadInvalidaException {
        if (edad < 0 || edad > 120) {
            throw new EdadInvalidaException("La edad debe estar entre 0 y 120.");
        }
    }
}
```

Salida en consola:

```
Output - Tp8-Interfaces-y-Excepciones (run)
run:
=== PARTE 2: EXCEPCIONES ===
Error: Divisi□n por cero.
Error: El texto no es un n□mero.
Error: El archivo no existe.
Edad inv□lida: La edad debe estar entre 0 y 120.
Error leyendo el archivo (try-with-resources).

BUILD SUCCESSFUL (total time: 0 seconds)
```

Tp8InterfacesYExepciones.java

```java
package tp8.interfaces.y.excepciones;

import Parte1.DemoParte1;
import Parte2.DemoParte2;

/**
 *
 * @author Ignacio Figueroa - TP 8 | Interfaces y Exepciones
 */
public class Tp8InterfacesYExcepciones {

    public static void main(String[] args) {
        DemoParte1.run();
        DemoParte2.run();
    }

}
```