Instituto Politécnico Nacional

Escuela Superior de Computo

Paradigmas de Programacion

Practica 12

Profesor: García Floriano Andrés

Integrantes:

Elizalde Hernández Alan

Reyes Ruiz Yoselyn Estefany

Solares Velasco Arturo Misael

Solís Lugo Mayra

Toral Hernández Leonardo Javier

3CV1

Introducción

Este reporte describe la implementación de un programa de cajero automático en Java. El cajero automático permite autenticar a los cuentahabientes, mostrar datos de cuenta y saldos, realizar depósitos en efectivo a cuentas propias y ajenas, transferir fondos entre cuentas y retirar efectivo. Además, se implementan excepciones personalizadas para manejar situaciones de saldo insuficiente en el cajero y en las cuentas.

Descripción del Código

En el código se hace el uso del nombre de los integrantes del equipo como nombres de usuario pertenecientes al cajero automático

N° de cuenta	Nombre del usuario	Contraseña
2023028447	Elizalde Hernández Alan	ELHLAN_g04
2023630858	Reyes Ruiz Yoselyn Estefany	RERLYN_a20
2023947596	Solares Velasco Arturo Misael	SOVTUR_q79
2023867455	Solís Lugo Mayra	SOLYRA_y45
2023385947	Toral Hernández Leonardo Javier	TOHRDO_m32

Clases y Métodos

1. Excepciones Personalizadas

SaldoEfectivoInsuficiente: Esta excepción se lanza cuando el saldo en efectivo del cajero es insuficiente para completar un retiro.

SaldoCuentaInsuficiente: Esta excepción se lanza cuando el saldo en la cuenta del usuario es insuficiente para completar una transferencia o retiro.

```
class SaldoEfectivoInsuficiente extends Exception {
    public SaldoEfectivoInsuficiente(String mensaje) {
        super(mensaje);
    }
}

class SaldoCuentaInsuficiente extends Exception {
    public SaldoCuentaInsuficiente(String mensaje) {
        super(mensaje);
    }
}
```

2. Clase Cuenta

Representa una cuenta bancaria con atributos para el número de cuenta, contraseña, nombre del titular y saldo. Incluye métodos para autenticar al usuario, obtener el saldo, realizar depósitos, retiros y transferencias

La clase Cuenta representa una cuenta bancaria con los siguientes atributos y métodos:

1. Atributos:
    - numeroCuenta: Número de cuenta del usuario.
    - password: Contraseña del usuario.
    - nombre: Nombre del titular de la cuenta.

- saldo: Saldo de la cuenta.

2. Métodos:

Constructor

```
public Cuenta(String numeroCuenta, String password, String nombre, double saldo) {
    this.numeroCuenta = numeroCuenta;
    this.password = password;
    this.nombre = nombre;
    this.saldo = saldo;
}
```

Inicializa los atributos de la cuenta.

Métodos Get

```
public String getNumeroCuenta() {
    return numeroCuenta;
}

public String getNombre() {
    return nombre;
}

public double getSaldo() {
    return saldo;
}
```

Permiten acceder a los atributos de la cuenta

Auntenticar

```
public boolean autenticar(String password) {
    return this.password.equals(password);
}
```

Verifica si la contraseña proporcionada coincide con la contraseña de la cuenta

Depositar

```
public void depositar(double monto) {
    this.saldo += monto;
}
```

Aumenta el saldo de la cuenta en el monto especificado

Retirar

```
public void retirar(double monto) throws SaldoCuentaInsuficiente {
    if (monto > this.saldo) {
        throw new SaldoCuentaInsuficiente("Saldo insuficiente en la cuenta.");
    }
    this.saldo -= monto;
}
```

Disminuye el saldo de la cuenta en el monto especificado. Lanza una excepción `SaldoCuentaInsuficiente` si el saldo es insuficiente

Transferir

```
public void transferir(Cuenta otraCuenta, double monto) throws SaldoCuentaInsuficiente {
    if (monto > this.saldo) {
        throw new SaldoCuentaInsuficiente("Saldo insuficiente en la cuenta para transferencia.");
    }
    this.saldo -= monto;
    otraCuenta.depositar(monto);
}
```

Transfiere un monto de esta cuenta a otra cuenta. Lanza una excepción `SaldoCuentaInsuficiente` si el saldo es insuficiente

3. Clase cajero automático

Maneja las operaciones del cajero, incluyendo la autenticación de usuarios, visualización de datos de cuenta, depósitos, transferencias y retiros. También mantiene el saldo de efectivo del cajero y una lista de cuentas

1. Atributos:

- saldoEfectivo: Saldo en efectivo disponible en el cajero.
- cuentas: Mapa de cuentas registradas en el cajero

2. Métodos

Constructor

```
public CajeroAutomatico(double saldoEfectivoInicial) {
    this.saldoEfectivo = saldoEfectivoInicial;
    this.cuentas = new HashMap<>();
}
```

Inicializa el saldo de efectivo del cajero y la colección de cuentas

- ### Agregar Cuenta

```
public void agregarCuenta(Cuenta cuenta) {
    cuentas.put(cuenta.getNumeroCuenta(), cuenta);
}
```

Agrega una nueva cuenta al mapa de cuentas del cajero

- ### Autenticar

```
public Cuenta autenticar(String numeroCuenta, String password) {
    Cuenta cuenta = cuentas.get(numeroCuenta);
    if (cuenta != null && cuenta.autenticar(password)) {
        return cuenta;
    }
    return null;
}
```

Verifica las credenciales del usuario y retorna la cuenta si la autenticación es exitosa

- ### Obtener Cuenta Por Número

```
public Cuenta obtenerCuentaPorNumero(String numeroCuenta) {
    return cuentas.get(numeroCuenta);
}
```

Retorna la cuenta asociada con el número de cuenta proporcionado

- ### Mostrar datos de la cuenta

```
public void mostrarDatosCuenta(Cuenta cuenta) {
    System.out.println("Número de cuenta: " + cuenta.getNumeroCuenta());
    System.out.println("Nombre: " + cuenta.getNombre());
    System.out.println("Saldo: " + cuenta.getSaldo());
}
```

Imprime los detalles de la cuenta

- ### Deposito en cuenta propia

```
public void depositarEnPropiaCuenta(Cuenta cuenta, double monto) {
    cuenta.depositar(monto);
}
```

Incrementa el saldo de la cuenta autenticada

Deposito en otra cuenta

```
public void depositarEnOtraCuenta(Cuenta cuentaDestino, double monto) {
    cuentaDestino.depositar(monto);
}
```

Incrementa el saldo de una cuenta destino

Transferir

```
public void transferir(Cuenta cuentaOrigen, Cuenta cuentaDestino, double monto) throws SaldoCuentaInsuficiente {
    cuentaOrigen.transferir(cuentaDestino, monto);
}
```

Transfiere fondos de una cuenta origen a una cuenta destino. Lanza una excepción SaldoCuentaInsuficiente si el saldo es insuficiente

Retiro en efectivo

```
public void retirarEfectivo(Cuenta cuenta, double monto) throws SaldoEfectivoInsuficiente, SaldoCuentaInsuficiente {
    if (monto > this.saldoEfectivo) {
        throw new SaldoEfectivoInsuficiente("Saldo insuficiente en el cajero.");
    }
    cuenta.retirar(monto);
    this.saldoEfectivo -= monto;
}
```

Disminuye el saldo de la cuenta autenticada y el saldo de efectivo del cajero. Lanza una excepción SaldoEfectivoInsuficiente si el saldo del cajero es insuficiente y SaldoCuentaInsuficiente si el saldo de la cuenta es insuficiente

4. Clase Main

Contiene el método main que implementa un menú para interactuar con el cajero automático. Los usuarios pueden autenticarse, consultar saldos, hacer depósitos, transferencias y retiros

- Se crea una instancia de `CajeroAutomatico` con un saldo inicial.
- Se agregan cuentas de ejemplo al cajero.
- Se muestra un menú de opciones para interactuar con el cajero.
- Las opciones del menú permiten autenticar al usuario, consultar saldos, realizar depósitos, transferencias y retiros.
- Se manejan excepciones para saldos insuficientes

Conclusión

El programa de cajero automático implementado en Java es una herramienta básica pero completa que permite gestionar cuentas bancarias y realizar operaciones financieras comunes. La estructura del código y el manejo de

excepciones personalizadas garantizan un funcionamiento robusto y seguro, proporcionando una base sólida para futuras extensiones y mejoras