CASO PRÁCTICO 6

- **TÍTULO: Programación orientada a objetos con Java**

- **SITUACIÓN**

  Tenemos que resolver los siguientes problemas para la empresa de programación para la que trabajamos.

- **INSTRUCCIONES**

  En el banco **Avaro Bank** se necesita desarrollar un sistema que gestione las cuentas de sus clientes y permita realizar operaciones básicas como depósitos y retiros de dinero. Para garantizar que todas las transacciones sean seguras y evitar errores en el manejo de los fondos, el sistema debe establecer ciertas reglas que regulen las operaciones y reaccionar adecuadamente ante intentos de realizar acciones inválidas.

  Cada **cuenta bancaria** está identificada por el **nombre de su titular** y tiene un **saldo inicial**. Para mantener la integridad de los datos, el **saldo de una cuenta nunca puede ser negativo**. Si en algún momento se intenta crear una cuenta con un saldo inicial menor a cero, el sistema debe impedirlo y generar una excepción **SaldoInvalidoException**.

  Los clientes podrán interactuar con el sistema a través de la entrada de datos por teclado. Al iniciar la aplicación, se pedirá al usuario que introduzca el nombre del titular de la cuenta y el saldo inicial. Posteriormente, el usuario podrá realizar depósitos y retiros introduciendo las cantidades correspondientes.

Para los depósitos, se exigirá que la cantidad a ingresar sea un valor positivo. Si alguien intenta hacer un depósito con una cantidad negativa o nula, la operación será rechazada y el sistema notificará que la transacción no es válida. Se lanzará una **CantidadInvalidaException**.

En cuanto a los retiros, solo podrán efectuarse si hay suficiente dinero disponible en la cuenta. Si un cliente intenta retirar una cantidad mayor al saldo disponible, el sistema impedirá la transacción y mostrará un mensaje de error informando que no hay fondos suficientes. Se lanzará una **FondosInsuficientesException**. Además, no se permitirá retirar cantidades negativas o iguales a cero, ya que estas operaciones no tienen sentido en el contexto bancario.

El usuario podrá realizar múltiples operaciones hasta que decida finalizar el programa. Para lograr un manejo adecuado de los errores y evitar fallos en el sistema, se utilizarán excepciones personalizadas que detecten y controlen cada tipo de operación inválida. En caso de que se produzca un error, el programa debe capturar la excepción y mostrar un mensaje claro al usuario sin interrumpir la ejecución del sistema.

Ejemplo de ejecución:

```
Introduzca el nombre del titular de la cuenta: Pepe
Introduzca el saldo inicial: 1200
Cuenta creada exitosamente.

Opciones: 1) Depositar 2) Retirar 3) Mostrar saldo 4) Salir
Seleccione una opción: 2
Introduzca la cantidad a retirar: 2000
Error en la operación: Fondos insuficientes. Saldo actual: 1200.0€

Opciones: 1) Depositar 2) Retirar 3) Mostrar saldo 4) Salir
Seleccione una opción: 1
Introduzca la cantidad a depositar: -20
Error en la operación: La cantidad a depositar debe ser mayor a cero.

Opciones: 1) Depositar 2) Retirar 3) Mostrar saldo 4) Salir
Seleccione una opción: 3

Titular: Pepe
Saldo actual: 1200,00€

Opciones: 1) Depositar 2) Retirar 3) Mostrar saldo 4) Salir
Seleccione una opción: 1
Introduzca la cantidad a depositar: 2000
Depósito realizado. Nuevo saldo: 3200,00€

Opciones: 1) Depositar 2) Retirar 3) Mostrar saldo 4) Salir
Seleccione una opción: 3

Titular: Pepe
Saldo actual: 3200,00€

Opciones: 1) Depositar 2) Retirar 3) Mostrar saldo 4) Salir
Seleccione una opción: 4
Saliendo del sistema. ¡Gracias por usar nuestro banco!
```