DIAGRAMA DE CLASES

ADSO - 2873711

Daniel Estiven
Arboleda Duque

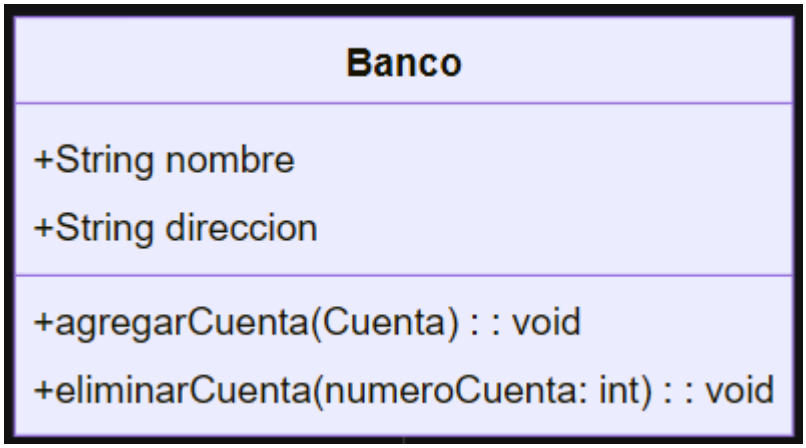
Contenido

2. ¿Qué métodos y clases usarías para registrar una transacción de retiro? 17

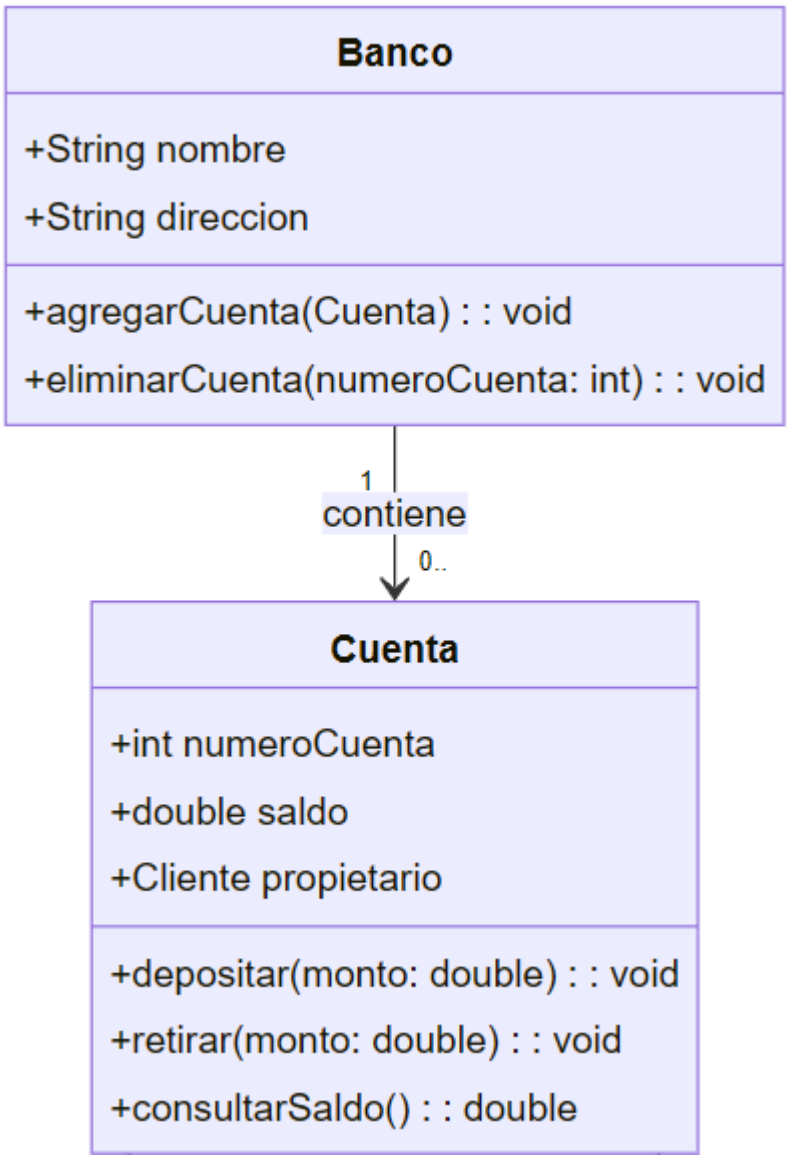
1. Ejercicio 1: Banco

- Define la clase Banco.
- Agrega los atributos nombre (String) y direccion (String).
- Agrega los métodos agregarCuenta(Cuenta): void y eliminarCuenta(numeroCuenta: int): void.
- Describe la relación entre Banco y Cuenta.

```
class Banco {
   +String nombre
   +String direccion
   +agregarCuenta(Cuenta): void
   +eliminarCuenta(numeroCuenta: int): void
}
```



La relación entre las clases banco y cuenta, es de, en un banco existen cuentas, y donde no exista un banco no existe una cuenta. En este caso el tipo de relación es de Agregación



Banco "1" --> "0.." Cuenta : contiene

Preguntas:

1. ¿Qué atributos necesita la clase Banco?

La clase banco necesita los atributos tipo String para nombre y tipo String para dirección.

2. ¿Qué métodos necesita la clase Banco?

Los métodos que necesita la clase banco son agregarCuenta(Cuenta): void y eliminarCuenta(numeroCuenta: int): void.

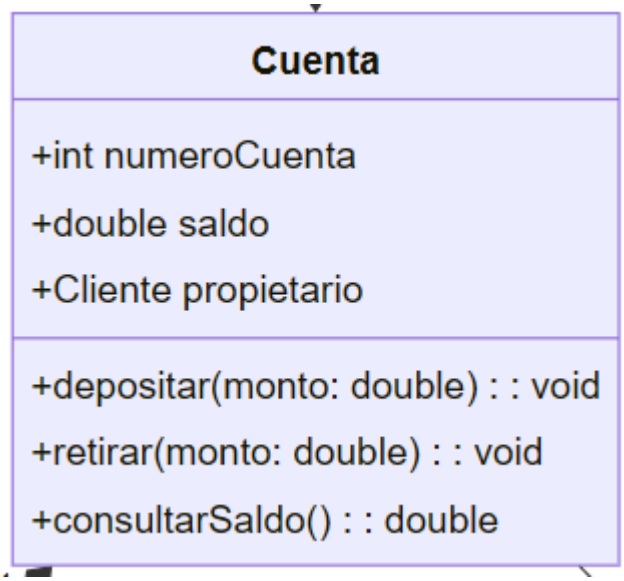
3. ¿Cómo describirías la relación entre Banco y Cuenta?

La clase cuenta depende de banco, pues con lógica, para que exista una cuenta, debe existir un banco al cual asociarla, si no existe un banco no seria posible hacer una cuenta.

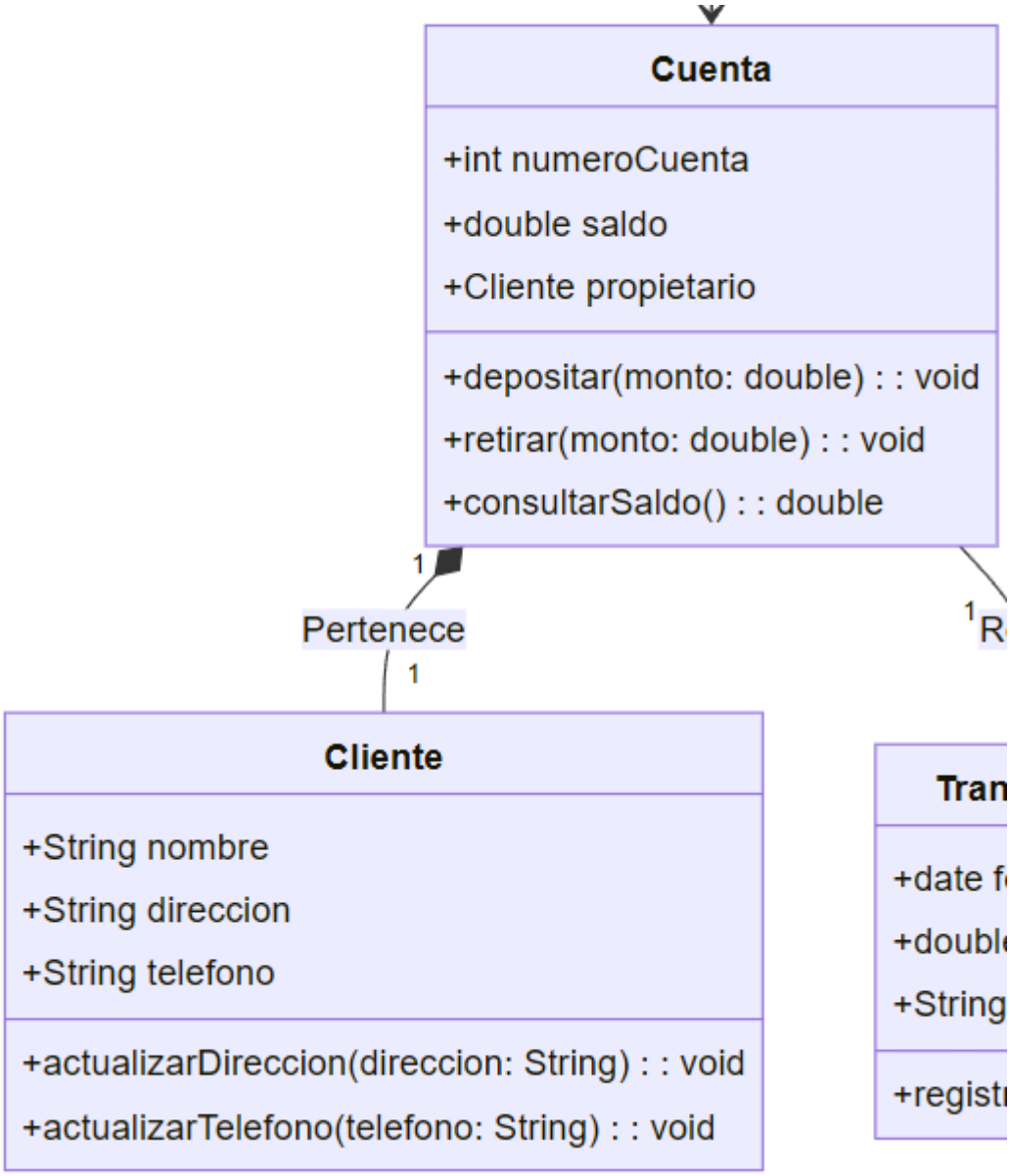
2. Ejercicio 2: Cuenta

- Define la clase Cuenta.
- Agrega los atributos numeroCuenta (int), saldo (double) y propietario (Cliente).
- Agrega los métodos depositar(monto: double): void, retirar(monto: double): void y consultarSaldo(): double.
- Describe la relación entre Cuenta y Cliente.

```
class Cuenta {
   +int numeroCuenta
   +double saldo
   +Cliente propietario
   +depositar(monto: double): void
   +retirar(monto: double): void
   +consultarSaldo(): double
}
```



La forma en la se relacionan las clases Cuenta y Cliente es que para que exista una cuenta en el banco, debe tener un usuario, un propietario, aquí es donde ingresa la clase cliente, que es donde se guarda los datos del usuario. En este caso el tipo de relación es de composición



Cuenta "1" *-- "1" Cliente : Pertenece

Preguntas:

1. ¿Qué atributos necesita la clase Cuenta?

Los atributos que necesita la clase Cuenta son: numeroCuenta tipo int, saldo tipo double y propietario tipo Cliente.

2. ¿Qué métodos necesita la clase Cuenta?

Los métodos que necesita la clase Cuenta son: depositar(monto: double): void, retirar(monto: double): void y consultarSaldo(): double.

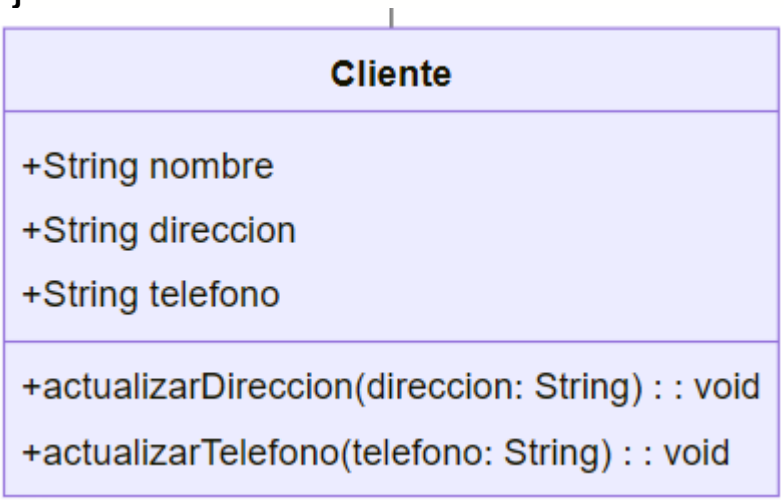
3. ¿Cómo describirías la relación entre Cuenta y Cliente?

La relación entre estas dos clases es que si no hay un cliente que cree una cuenta, no puede existir una cuenta en la base del banco, por ende, aunque la relación sea de tipo composición la cuenta depende de cliente.

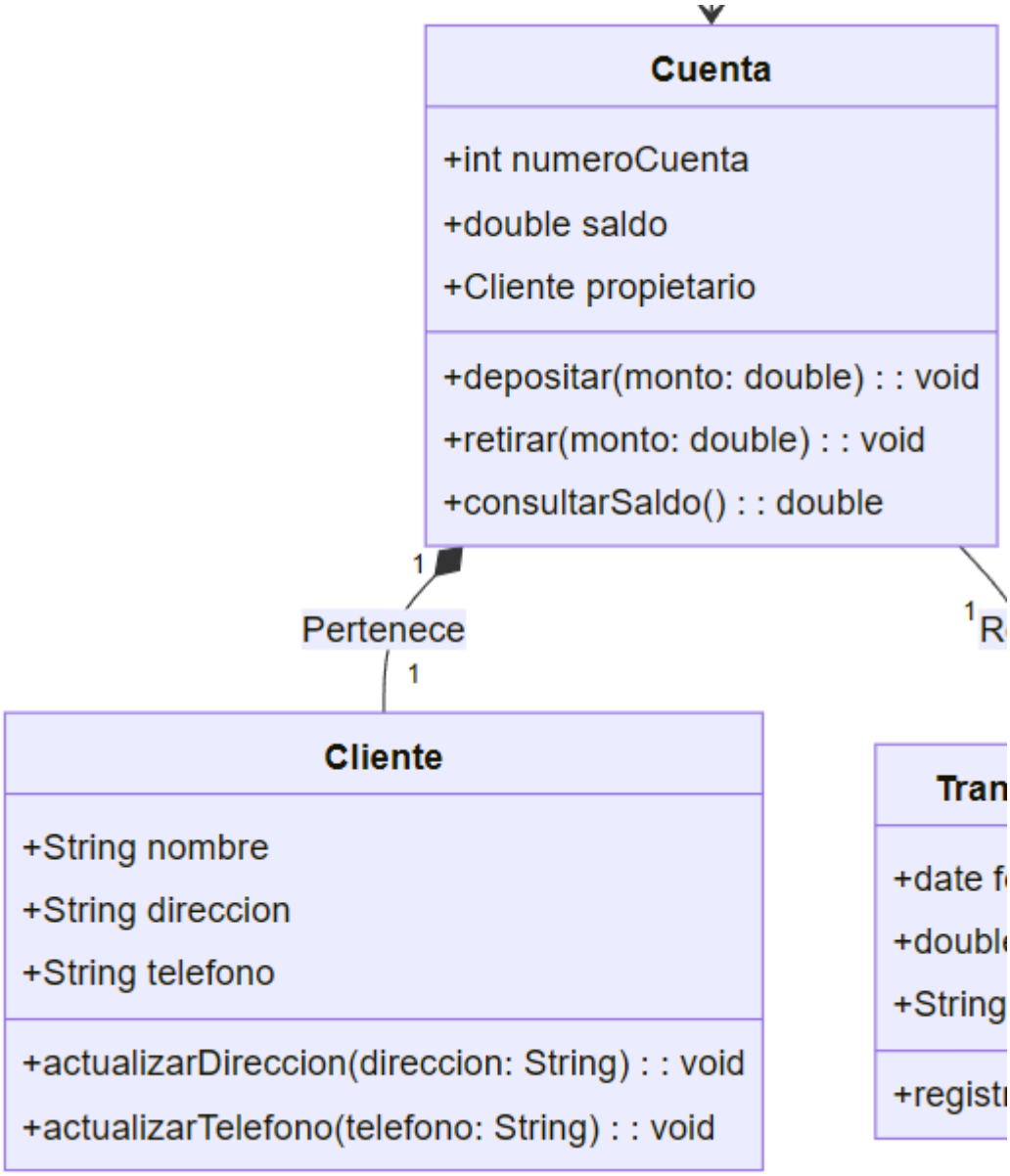
3. Ejercicio 3: Cliente

- Define la clase Cliente.
- Agrega los atributos nombre (String), direccion (String) y telefono (String).
- Agrega los métodos actualizarDireccion(direccion: String): void y actualizarTelefono(telefono: String): void.
- Describe cómo un Cliente puede interactuar con una Cuenta.

```
class Cliente {
+String nombre
+String direccion
+String telefono
+actualizarDireccion(direccion: String): void
+actualizarTelefono(telefono: String): void
}
```



La forma en la que interactúan las clases Cliente y Cuenta, es que, cliente tiene una cuenta en la cual en la clase cuenta se tiene guardada una variable de tipo cliente, para que en la clase cuenta se vea el nombre y datos del usuario creado en la clase Cliente. El tipo de relación es de composición.



Cuenta "1" *-- "1" Cliente : Pertenece

Preguntas:

1. ¿Qué atributos necesita la clase Cliente?

Los atributos que necesita la clase Cliente nombre tipo String, dirección tipo String y telefono tipo String.

2. ¿Qué métodos necesita la clase Cliente?

Los métodos que necesita la clase Cliente son: actualizarDireccion(direccion: String): void y actualizarTelefono(telefono: String): void.

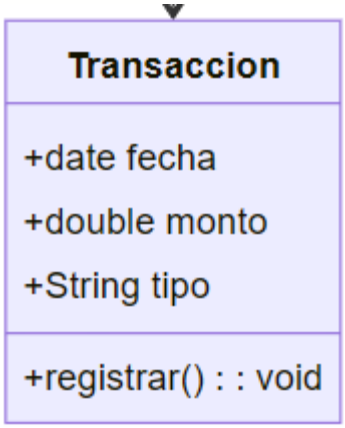
3. ¿Cómo describirías la interacción entre Cliente y Cuenta?

La interacción entre las dos clases es de tipo composición como ya se mostró anteriormente, pues la en la clase Cliente se crea un usuario y para poder crear una cuenta, la clase cuenta debe tener los datos del usuario, es decir, la Clase cuenta depende de la clase Cliente para poder realizar los procesos que tiene.

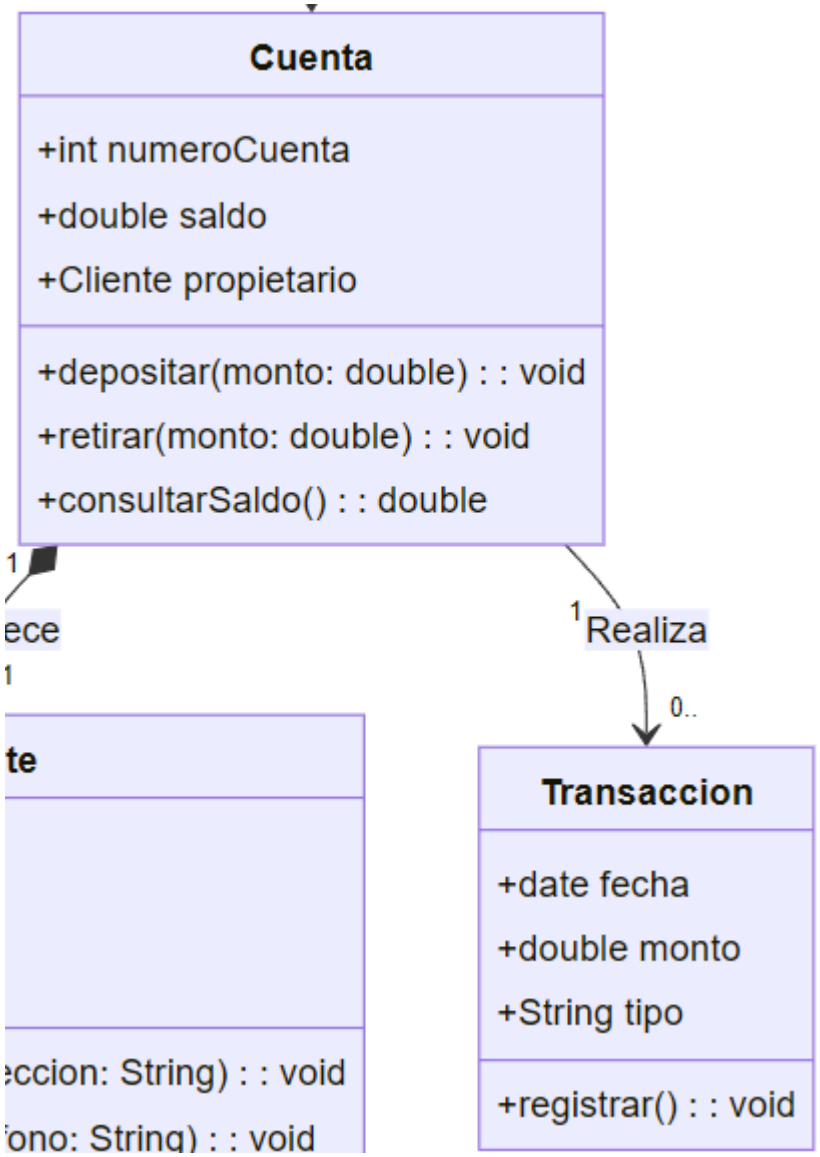
4. Ejercicio 4: Transacción

- Define la clase Transacción.
- Agrega los atributos fecha (Date), monto (double) y tipo (String).
- Agrega el método registrar (): void.
- Describe la relación entre Cuenta y Transacción.

```
class Transaccion {
   +date fecha
   +double monto
   +String tipo
   +registrar(): void
}
```



La relación entre estas dos clases es de la forma en la que la clase Transaccion depende de la clase cuenta, pues para realizar una transacción, se necesita un valor y la cuenta de donde saldrá, y estos datos lo guarda la clase Cuenta, por ende una cuenta puede realizar muchas transacciones pero la transacción solo sale de una cuenta siendo una relación de tipo Asignación.



Cuenta "1" **-->** "0.." Transaccion **:** Realiza

Preguntas:

1. ¿Qué atributos necesita la clase Transacción?

Los atributos que necesita la clase Transacción son: fecha tipo Date, monto tipo double y tipo de tipo String.

2. ¿Qué métodos necesita la clase Transacción?

Los métodos que necesita la clase son: registrar (): void.

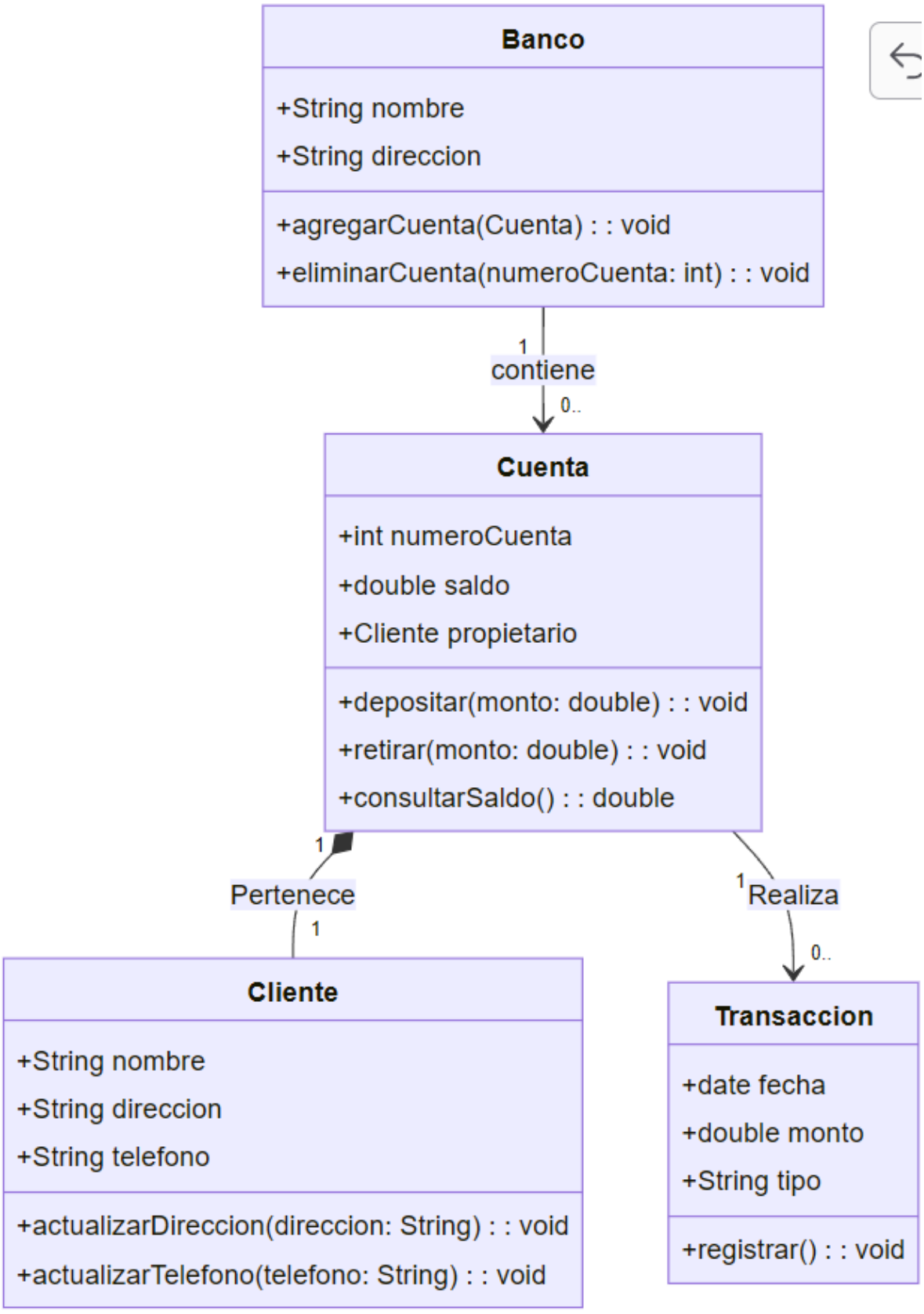
3. ¿Cómo describirías la relación entre Cuenta y Transacción?

La relación entre las dos clases es de tipo asignación pues la clase Cuenta puede realizar muchas transacciones y la clase Transaccion solo hace el proceso de una sola cuenta.

5. Ejercicio 5: Integración

- Dibuja un diagrama de clases que incluya Banco, Cuenta, Cliente y Transaccion.
- Define las relaciones entre estas clases según los ejercicios anteriores.
- Explica cómo se relacionan estas clases en un sistema bancario.

```
class Banco {
    +String nombre
    +String direccion
    +agregarCuenta(Cuenta): void
    +eliminarCuenta(numeroCuenta: int): void
}
class Cuenta {
    +int numeroCuenta
    +double saldo
    +Cliente propietario
    +depositar(monto: double): void
    +retirar(monto: double): void
    +consultarSaldo(): double
}
    class Cliente {
    +String nombre
    +String direccion
    +String telefono
    +actualizarDireccion(direccion: String): void
    +actualizarTelefono(telefono: String): void
}
class Transaccion {
    +date fecha
    +double monto
    +String tipo
    +registrar(): void
}
```


Banco
+String nombre
+String direccion
+agregarCuenta(Cuenta) : : void
+eliminarCuenta(numeroCuenta: int) : : void
1
contiene
0..
Cuenta
+int numeroCuenta
+double saldo
+Cliente propietario
+depositar(monto: double) : : void
+retirar(monto: double) : : void
+consultarSaldo() : : double
1
Pertenece
1
1
Realiza
0..
Cliente
+String nombre
+String direccion
+String telefono
+actualizarDireccion(direccion: String) : : void
+actualizarTelefono(telefono: String) : : void
Transaccion
+date fecha
+double monto
+String tipo
+registrar() : : void

Las relaciones son:

De Banco a Cuenta es de 1 a muchos Banco "1" --> "0.." Cuenta : contiene

de Cliente a Cuenta es de 1 a 1 Cuenta "1" *-- "1" Cliente : Pertenece

de Cuenta a Transaccion es de 1 a muchos Cuenta "1" --> "0.." Transaccion : Realiza

Los tipos de relación son:

La relación entre Banco y Cuenta es de tipo Asignación, la relación entre Cliente y Cuenta es de tipo Composición y la relación entre Cuenta y Transaccion es de tipo Asignación. Esto convirtiendo que Para que en una transacción se pueda realizar debe existir una cuenta, para que la cuenta exista debe tener datos del propietario, para esto se necesita a el cliente y para que exista un cliente con una cuenta y pueda realizar transacción debe existir un Banco donde se tengan todos estos datos y se puedan realizar todos los procesos.

Preguntas:

1. ¿Cómo se relacionan Banco y Cuenta en el diagrama de clases?

La forma en la que se relacionan es con una relación de tipo Asignación, significando que en un banco existen muchas cuentas, pero una cuenta solo pertenece a un Banco.

2. ¿Cómo se relacionan Cuenta y Cliente en el diagrama de clases?

La forma en la que se relacionan es con una relación de tipo composición, significando que un cliente solo puede tener una cuenta y una cuenta solo puede tener un solo propietario es decir un cliente.

3. ¿Cómo se relacionan Cuenta y Transaccion en el diagrama de clases?

La forma en la que se relacionan es con una relación de tipo asignación, significando que una cuenta puede realizar muchas transacciones, pero una transacción solo sale de una cuenta.

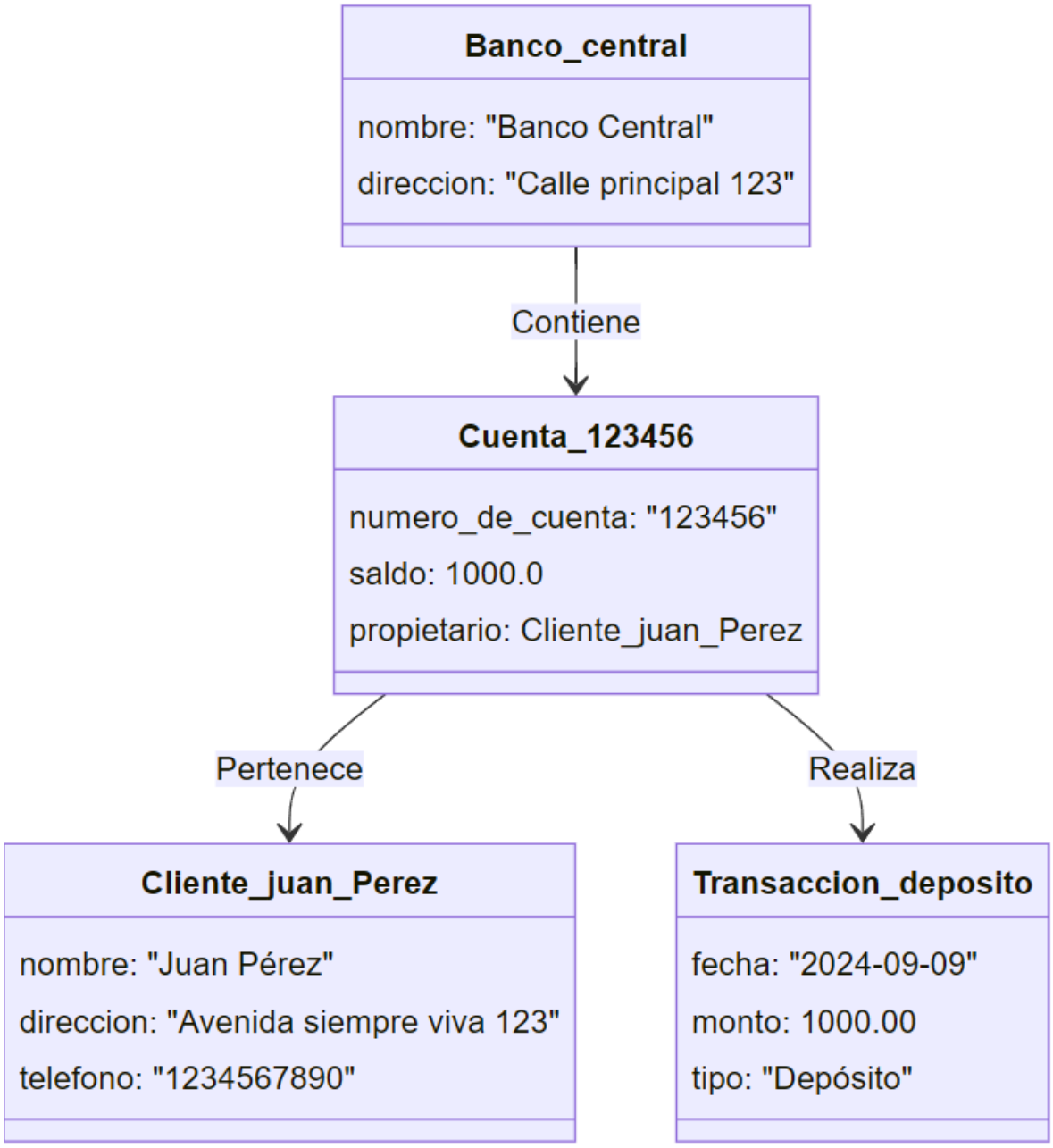
6. Ejercicio 6: Escenarios de Uso

Escenario 1:

- Un cliente llamado Juan Pérez desea abrir una nueva cuenta en el banco "Banco Central".
- Juan realiza un depósito inicial de $1000.00 en su nueva cuenta.
- Diseña las clases y los métodos necesarios para manejar este escenario.

```
class Banco_central {
  nombre: "Banco Central"
```

```
        direccion: "Calle principal 123"
    }
    class Cuenta_123456 {
        numero_de_cuenta: "123456"
        saldo: 1000.0
        propietario: Cliente_juan_Perez
    }
    class Cliente_juan_Perez {
        nombre: "Juan Pérez"
        direccion: "Avenida siempre viva 123"
        telefono: "1234567890"
    }
    class Transaccion_deposito {
        fecha: "2024-09-09"
        monto: 1000.00
        tipo: "Depósito"
    }
Banco_central --> Cuenta_123456 : Contiene
Cuenta_123456 --> Cliente_juan_Perez : Pertenece
Cuenta_123456 --> Transaccion_deposito : Realiza
```

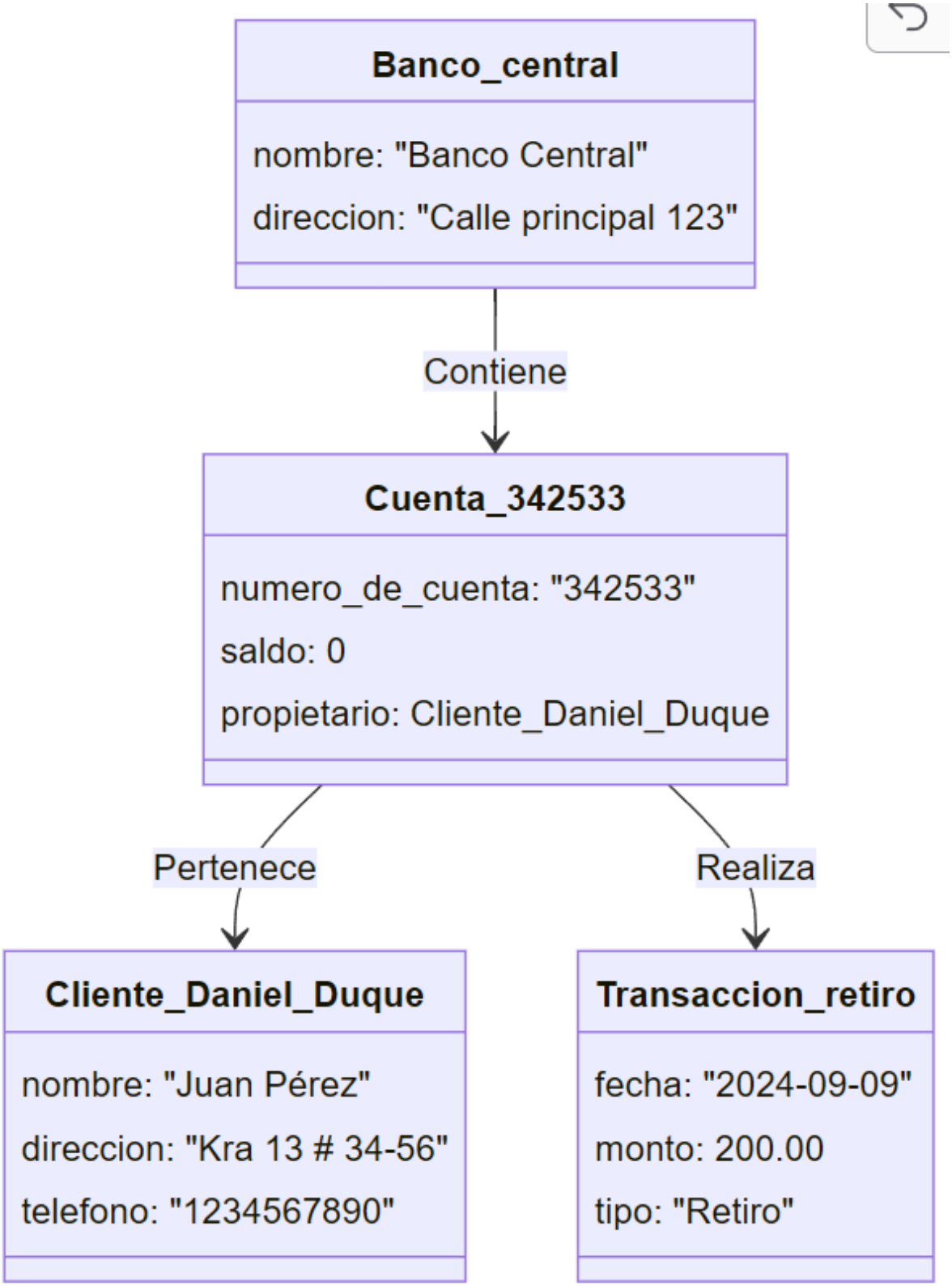


Escenario 2:

- Un cliente realiza una transacción de retiro de $200.00.
- La transacción debe registrarse en la cuenta del cliente.
- Diseña las clases y los métodos necesarios para manejar este escenario.

```
class Banco_central {
   nombre: "Banco Central"
```

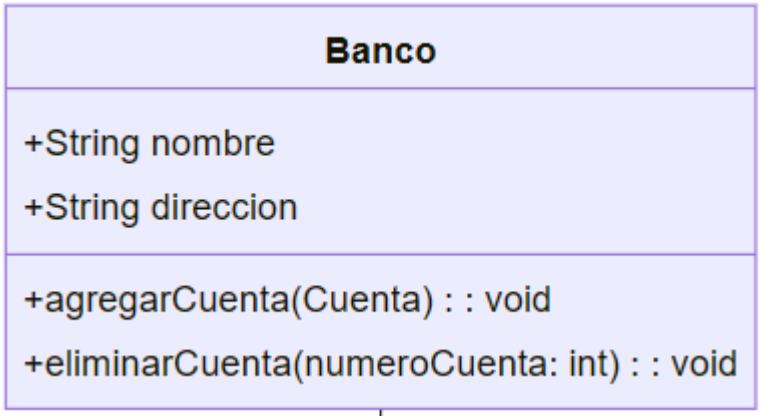
```
        direccion: "Calle principal 123"
    }
    class Cuenta_342533 {
        numero_de_cuenta: "342533"
        saldo: 0
        propietario: Cliente_Daniel_Duque
    }
    class Cliente_Daniel_Duque {
        nombre: "Juan Pérez"
        direccion: "Kra 13 # 34-56"
        telefono: "1234567890"
    }
    class Transaccion_retiro {
        fecha: "2024-09-09"
        monto: 200.00
        tipo: "Retiro"
    }
Banco_central --> Cuenta_342533 : Contiene
Cuenta_342533 --> Cliente_Daniel_Duque : Pertenece
Cuenta_342533 --> Transaccion_retiro : Realiza
```


Banco_central
nombre: "Banco Central"
direccion: "Calle principal 123"
Contiene
Cuenta_342533
numero_de_cuenta: "342533"
saldo: 0
propietario: Cliente_Daniel_Duque
Pertenece
Realiza
Cliente_Daniel_Duque
nombre: "Juan Pérez"
direccion: "Kra 13 # 34-56"
telefono: "1234567890"
Transaccion_retiro
fecha: "2024-09-09"
monto: 200.00
tipo: "Retiro"

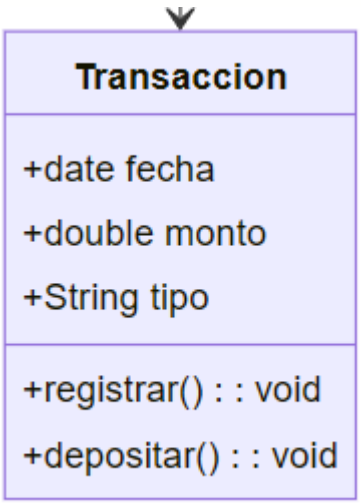
Preguntas:

1. ¿Qué métodos y clases usarías para abrir una cuenta y realizar un depósito?

Para abrir una cuenta usaría la clase Banco para utilizar el método +agregarCuenta**(**Cuenta**):** void.

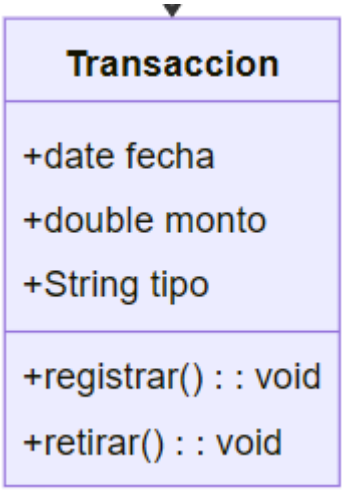


Para realizar un Depósito usaría la clase Transaccion con el método **+**registrar**():** void y **+**depositar**():** void



2. ¿Qué métodos y clases usarías para registrar una transacción de retiro?

Para registrar una transacción de retiro, usaría la clase transacción con los métodos **+**registrar**():** void y **+**retirar**():** void


Transaccion
+date fecha
+double monto
+String tipo
+registrar() : : void
+retirar() : : void