

Crear un programa que simule una cuenta bancaria, con las siguientes características:

- Crear una clase "CuentaBancaria" con los campos y métodos correspondientes \*ver diagrama UML adjunto en la imagen\*
- El primer método que vemos en el diagrama es el constructor de la clase, que recibirá los argumentos que ahí se indican e inicializará a los campos de la clase.
- El método "Deposito" se encarga de recibir una suma de dinero desde el exterior y la agregará al campo "saldo".
- El método "Retiro" se encarga de extraer una suma de dinero del campo "saldo" dada desde el exterior (verificar que sea posible retirar la cantidad de dinero a partir del saldo que ya se encuentra en la cuenta.
- El método "ConsultaSaldo" se encarga simplemente de mostrarnos el dinero que tenemos en nuestra cuenta bancaria.
- El método "ToString" se encarga de mostrarnos la información del cliente (objeto).

En Main nos encargaremos de pedir los siguientes datos:

- nombre, apellidos, dirección, rfc(en México el RFC es una clave que nos permite cumplir con nuestras obligaciones fiscales, ustedes pueden usar el similar de su país) y un saldo inicial para depositar a la cuenta bancaria.
- Todos esos datos serán enviados al constructor de la clase "CuentaBancaria" en el momento de instanciarla.
- Después crearemos un menú que aparecerá mientras el usuario no decida salir (apoyarse de un Do-While), y contendrá las siguiente opciones:

0. Deposito

1. Retiro

2. Consultar Saldo

3. Mostrar información de la cuenta

4. Salir

- Dependiendo de la opción que se escoja, mandar a llamar al método correspondiente de la clase y crear su lógica para que se pueda depositar o retirar dinero.

**\*NOTA\*** Es una tarea algo amplia en la que puede que surjan algunos problemas y preguntas, intenten resolverlo antes de ver mi vídeo de resolución, les deseo mucha suerte. Les dejo también el diagrama del método "Deposito" para que no estén tan perdidos.

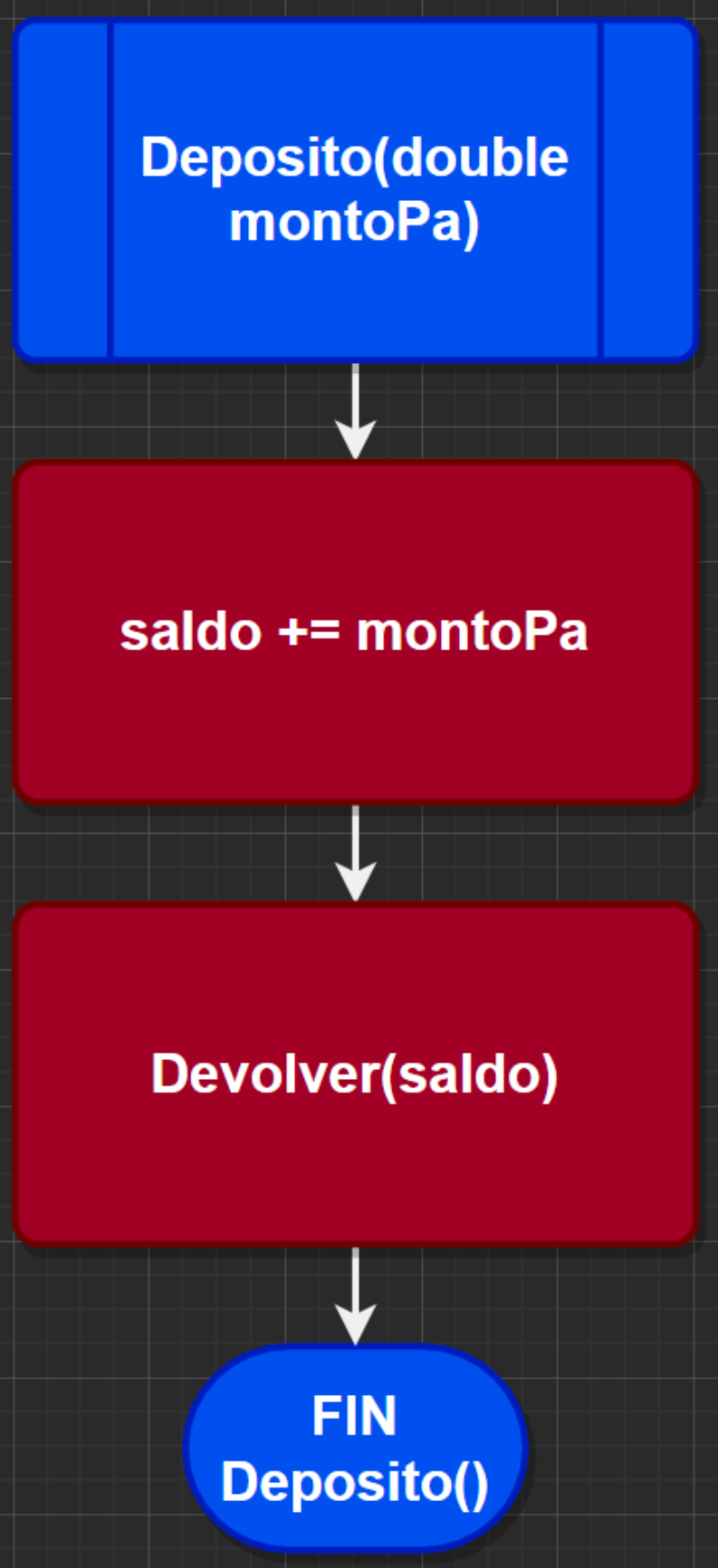


## CuentaBancaria

- nombre: string
- apellidos: string
- direccion: string
- rfc: string
- saldo: double

- + CuentaBancaria(nombrePa, apellidosPa, saldoPa, direccionPa, rfcPa)
- + Deposito(montoPa) : double
- + Retiro(montoPa) : double
- + ConsultaSaldo() : void
- + ToString : string

**Deposito(double  
montoPa)**



```
graph TD; A[Deposito(double montoPa)] --> B[saldo += montoPa]; B --> C[Devolver(saldo)]; C --> D([FIN Deposito()]);
```

The flowchart illustrates the execution of the Deposito method. It begins with a blue rectangular box containing the method signature 'Deposito(double montoPa)'. A white arrow points down to a red rectangular box with the operation 'saldo += montoPa'. Another white arrow points down to a second red rectangular box containing 'Devolver(saldo)'. A final white arrow points down to a blue oval terminal box labeled 'FIN Deposito()'.

**saldo += montoPa**

**Devolver(saldo)**

**FIN  
Deposito()**