**Descripción general**

Este proyecto busca implementar una simulación de un banco con énfasis en la programación orientada a objetos. Para lograr ésto, se creó una aplicación de escritorio en la cual existen varios clientes, cada uno con sus respectivas tarjetas bancarias, los cuales pueden libremente hacer transacciones entre ellos como deseen. Además, existen facturas, las cuales requieren un procedimiento específico para ser pagadas, y divisas (asociadas a cada tarjeta), las cuales imponen algunas restricciones sobre cuáles transacciones son permitidas. En este programa se implementaron 5 funcionalidades, que posibilitan hacer transacciones, pagar facturas, convertir de una divisa a otra, deshacer una transacción, entre otras.

**Descripción del diseño estático del sistema**

**Implementación de características de POO**

**Clase y métodos abstractos:**

Como clase, implementamos la clase abstracta tarjeta en la línea 14 del archivo Tarjeta.java:



También implementamos algunos métodos abstractos, así como la función tieneSaldo() en la línea 121 del archivo Tarjeta.java:



**Interfaz:**

En este proyecto, creamos una interfaz llamada BorrarTarjeta. Esta se declara en la línea 3 del archivoBorrarTarjeta.java:



**Herencia**

Existen en nuestro proyecto dos clases que heredan de nuestra clase abstracta Tarjeta: TarjetaDebito, y TarjetaCredito: (En los archivos TarjetaDebito.java y TarjetaCredito.java)





**Ligadura dinámica**

Utilizamos ligadura dinámica en algunos puntos en este proyecto, por ejemplo, en los siguientes métodos abstractos de la clase Tarjeta: (Archivo Tarjeta.java, líneas 107 y 115)





Estos métodos son casos de ligadura dinámica, ya que en tiempo de compilación, el computador no sabe si implementar el método definido en TarjetaDebito o el método definido en TarjetaCredito.

**Atributos de clase y métodos de clase**

En este proyecto utilizamos varios atributos de clase. Por ejemplo, la clase Transaccion tiene el atributo transacciones: (Línea 34 del archivo Transaccion.java)



Además tenemos algunos métodos de clase, por ejemplo, el método modificarPuntaje() de la clase Factura: (Línea 93 del archivo Factura.java)



**Uso de constante**

Usamos algunas constantes en este programa, por ejemplo, el atributo NOMBRE de la clase cliente: (Línea 21 archivo Cliente.java)



**Encapsulamiento**

En la clase Cliente utilizamos un atributo público:(Línea 21 archivo Cliente.java)



En la clase Tarjeta utilizamos un atributo protegido: (Línea 18 del archivo Tarjeta.java



Y finalmente, la clase Canal utiliza el siguiente atributo privado: (Línea 29 del archivo Canal.java)



**Sobrecarga de métodos y constructores**

Utilizamos varios casos de sobrecarga de métodos, por ejemplo, en la clase Transacción, existen estas tres versiones de encontrarTransacciones():(Archivo Transaccion.java, líneas 189, 199 y 209)







**Manejo de this y this()**

Utilizamos el this para desambiguar en algunos constructores, por ejemplo, en uno de los constructores de Transaccion: (Línea 38 del archivo Transaccion.java)



También usamos this en el constructor de Cliente: (Línea 30 del archivo Cliente.java)



El this() lo utilizamos en dos constructores de Transaccion, por ejemplo: (Línea 75 del archivo Transaccion.java)



También lo usamos en un constructor de Canal: (Línea 40 del archivo Canal.java)



**Implementación de enum**

Implementamos un enum Divisa: (Línea 13 del archivo Divisa.java)



**Descripción de las funcionalidades**

**PagarFactura (Opción número 1)**

La función `pagarFactura()` implementa un proceso de pago de una factura. Aquí está el resumen de los objetos que intervienen en su implementación:

1. `Cliente`: Representa un cliente que realizará el pago de la factura. Se selecciona a través de la función `elejirUsuario()`.

2. `Factura`: Es la factura que se pagará. El cliente seleccionado tiene una lista de facturas, y se muestra al usuario una lista de facturas disponibles para pagar. El usuario elige una opción y se obtiene la factura correspondiente.

3. `Tarjeta`: Representa una tarjeta que el cliente puede utilizar para el pago. Se obtiene una lista de tarjetas posibles a partir de las tarjetas asociadas al cliente y la factura seleccionada.

4. `Transaccion`: Representa la transacción generada para realizar el pago de la factura. Se genera utilizando la factura, el monto a transferir y la tarjeta seleccionada.

***el proceso que se hace es el siguiente:***

Seleccionar cliente -> Verificar si se seleccionó un cliente válido -> Verificar si el cliente tiene facturas pendientes (si no tiene, se le permite al usuario escoger otro cliente) -> Mostrar las facturas disponibles para pagar -> Escoge la factura que deseas pagar -> mostrar las tarjetas disponibles para pagar -> Escoge la tarjeta que quieres utilizar para pagar -> Obtener el monto a transferir -> Generar la transacción -> Verificar si la transacción fue rechazada (si es rechazada decir por que) -> Verificar si se debe eliminar la tarjeta -> Confirmar si se desea continuar con el proceso de pago (en caso de que no, se cancela el pago) -> Realizar el pago de la factura -> Reemplazar la factura anterior con la factura nueva -> Restablecer los errores de la tarjeta a cero

**Cambiar divisa (Opción número 2)**

La función `cambiarDivisa()` implementa la conversión de divisas y utiliza los siguientes objetos:

1. `Cliente`: Representa al cliente que realiza la operación.

2. `Divisa`: Enumeración que define las diferentes divisas disponibles.

3. `Tarjeta`: Representa una tarjeta asociada al cliente, con una divisa específica.

4. `TarjetaCredito`: Subclase de `Tarjeta` que representa una tarjeta de crédito.

5. `ArrayList<Divisa>`: Lista de divisas utilizadas para verificar las tarjetas disponibles y las divisas seleccionadas.

6. `ArrayList<Tarjeta>`: Lista de tarjetas disponibles para la conversión de divisas.

7. `ArrayList<Tarjeta>`: Lista de tarjetas seleccionadas para la conversión (una tarjeta de origen y otra de destino).

8. `Canal`: Representa el canal a través del cual se realiza la conversión de divisas.

9. `ArrayList<Canal>`: Lista de canales disponibles para el cliente.

10. `Transaccion`: Representa una transacción de conversión de divisas.

En resumen, la función `cambiarDivisa()` permite al cliente seleccionar las divisas de origen y destino, elegir las tarjetas correspondientes, seleccionar el canal de conversión y el monto a convertir. Luego, realiza la conversión de divisas y genera una transacción que registra los detalles de la operación. Finalmente, se muestra la transacción resultante al usuario.

***el proceso que se hace es el siguiente:***

Seleccionar cliente -> Verificar si se seleccionó un cliente válido -> Escoger la divisa de origen -> Obtener el número de la divisa de origen seleccionada -> Validar la opción seleccionada -> Almacenar la referencia de la divisa de origen seleccionada -> Escoger la divisa de destino -> Almacenar la referencia de la divisa de destino seleccionada -> Escoger la divisa de destino -> Obtener el número de la divisa de destino seleccionada -> Validar la opción seleccionada -> Almacenar la referencia de la divisa de destino seleccionada -> Verificar si se seleccionó la misma divisa de origen y destino -> Verificar si el cliente tiene tarjetas correspondientes a las divisas seleccionadas -> Escoger la tarjeta de origen -> Obtener la opción de tarjeta de origen seleccionada -> Verificar que la tarjeta de origen seleccionada tenga la divisa correspondiente -> Escoger la tarjeta de destino -> Obtener la opción de tarjeta de destino seleccionada -> Verificar que la tarjeta de destino seleccionada tenga la divisa correspondiente -> Listar los canales disponibles para el cliente -> Obtener la opción de canal seleccionada -> Solicitar y validar el monto a convertir -> Realizar la conversión de divisas -> Crear la transacción -> Crear la transacción -> Finalizar la conversión a través del canal seleccionado

**RetirarODepositar (Opción número 3)**

En la implementación de la función "retirarOrDepositar()", los siguientes objetos intervienen:

1. Cliente: Representa un cliente del banco y se utiliza para seleccionar al cliente que realizará la operación de retiro o depósito.

2. Divisa: Representa una divisa disponible para el cliente y se utiliza para mostrar las opciones de divisas disponibles y seleccionar la divisa con la que se realizará la operación.

3. Tarjeta: Representa una tarjeta asociada al cliente y a una divisa específica. Se utilizan para mostrar las opciones de tarjetas disponibles y seleccionar la tarjeta con la que se realizará la operación.

4. Canal: Representa un canal a través del cual se puede realizar la operación. Se utilizan para mostrar las opciones de canales disponibles y seleccionar el canal con el que se realizará la operación.

5. Transaccion: Representa una transacción bancaria y se utiliza para crear la transacción inicial y finalizarla. Se utiliza para almacenar información sobre la transacción generada y mostrarla al usuario.

Estos objetos interactúan entre sí para permitir al cliente seleccionar la divisa, tarjeta y canal apropiados para realizar la operación de retiro o depósito, y finalmente realizar la transacción bancaria correspondiente.

***el proceso que se hace es el siguiente:***

Obtener el cliente seleccionado (Si no se seleccionó ningún cliente, finalizar la función) -> Mostrar opciones al usuario -> Leer la opción seleccionada por el usuario -> Bucle para validar la opción seleccionada -> Verificar si se realizará un retiro o un depósito -> Obtener las divisas disponibles para el cliente -> Verificar si el cliente tiene alguna divisa disponible -> Mostrar las opciones de divisas disponibles al cliente -> Leer la elección de la divisa -> Validar la elección de la divisa -> Obtener la divisa seleccionada -> Obtener las tarjetas disponibles para la divisa y la operación seleccionada -> Verificar si el cliente tiene alguna tarjeta disponible -> Mostrar las opciones de tarjetas disponibles al cliente -> Leer la elección de la tarjeta -> Validar la elección de la tarjeta -> Obtener la tarjeta seleccionada -> Obtener los canales disponibles para la divisa y la operación seleccionada -> Verificar si hay algún canal disponible -> Mostrar las opciones de canales disponibles al cliente -> Leer la elección del canal -> Validar la elección del canal -> Obtener el canal seleccionado -> Solicitar el monto de la operación al cliente -> Crear la transacción inicial -> Mostrar información de la transacción inicial generada -> Verificar si la transacción inicial fue rechazada (y decir por que, en este caso, se rompe el proceso) -> Solicitar confirmación al cliente -> Validar la opción seleccionada por el cliente (Si el cliente no desea continuar, finalizar el proceso) -> Finalizar la transacción -> Mostrar mensaje de éxito y la información de la transacción final

**solicitarTarjetaCredito (Opción número 4)**

En la implementación de la función "solicitarTarjetaCredito()", los siguientes objetos intervienen:

1. Cliente: Representa un cliente del banco y se utiliza para seleccionar al cliente que solicitará la tarjeta de crédito.

2. Transaccion: Representa una transacción bancaria y se utiliza para revisar el historial de transacciones de crédito del cliente y calcular un puntaje tentativo.

3. Tarjeta: Representa una tarjeta bancaria y se utiliza para obtener las tarjetas bloqueadas y activas del cliente.

4. Banco: Representa el banco y se utiliza para calcular el puntaje definitivo del cliente y seleccionar las tarjetas de crédito disponibles.

5. Divisa: Enumeración que representa las diferentes divisas disponibles para la tarjeta de crédito y se utiliza para mostrar las opciones de divisas disponibles.

6. TarjetaCredito: Representa una tarjeta de crédito y se utiliza para obtener las tarjetas de crédito disponibles según el puntaje definitivo y la divisa seleccionada, y para añadir una tarjeta de crédito al cliente.

7. Factura: Representa una factura y se utiliza para modificar el puntaje del cliente en función de las tarjetas bloqueadas y activas.

Estos objetos interactúan entre sí para permitir al cliente seleccionar la divisa, la tarjeta de crédito y realizar la solicitud correspondiente. Se revisa el historial de créditos del cliente, se calcula un puntaje tentativo y se ajusta el puntaje definitivo en base a las tarjetas bloqueadas y activas. Luego, se muestran las opciones de divisas y tarjetas de crédito disponibles para que el cliente realice su elección. Finalmente, se añade la tarjeta de crédito seleccionada al cliente.

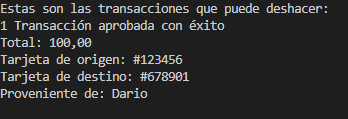
Aquí están los comentarios del código proporcionado:

1. Seleccionar un cliente -> Verificar si el cliente es nulo -> Obtener el historial de transferencias del cliente -> Calcular el puntaje tentativo del cliente en base al historial de transferencias -> Obtener las tarjetas de crédito bloqueadas del cliente -> Obtener las tarjetas de crédito activas (no bloqueadas) del cliente -> Modificar el puntaje definitivo del cliente en base a las tarjetas bloqueadas y activas -> Mostrar las opciones de divisas disponibles al cliente -> Leer la elección de la divisa -> Obtener la divisa seleccionada -> Obtener las tarjetas de crédito disponibles en base al puntaje definitivo y la divisa seleccionada -> Mostrar puntaje definitivo -> Mostrar las tarjetas de crédito disponibles con sus respectivos puntos y bonos -> Añadir la tarjeta de crédito seleccionada al cliente con el bono correspondiente

**Deshacer transacción (Opción número 5)**

Esta funcionalidad sirve para deshacer una transacción hecha de un cliente a otro. Al inicio, el usuario puede escoger mediante cuál criterio desea filtrar las transacciones. Existen tres criterios posibles: La divisa de la transacción, la tarjeta usada en la transacción, o el cliente que recibió la transacción. Después de que el cliente haya hecho esta decisión, una función estática de la clase Transacción (llada encontrarTransacciones()) se encarga de encontrar las transacciones que correspondan con el criterio del usuario.

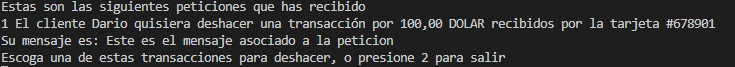
Ejemplo: Lista de transacciones con 1 elemento:



El usuario escoge cuál transacción desea deshacer, y tendrá la oportunidad de ingresar un mensaje, el cual será visto por el cliente que recibió la transacción. Después de ésto, el programa automáticamente vuelve al menú inicial.

Un método estático de la clase Banco (llamado generarPeticion()) se encarga de generar una transacción pendiente con el mensaje ingresado por el usuario. En este punto, aún no se ha transferido dinero de una tarjeta a otra, ya que el sistema está esperando la luz verde por parte del usuario que recibió la transacción. Para el último paso de esta funcionalidad, es necesario que el usuario vuelva al menú principal y escoja la opción 6 (ver peticiones). Esta opción inicialmente le mostrará todas peticiones que hayan hecho otros clientes al cliente que está usando el programa, de la siguiente forma:

Ejemplo: Lista de peticiones con 1 elemento:



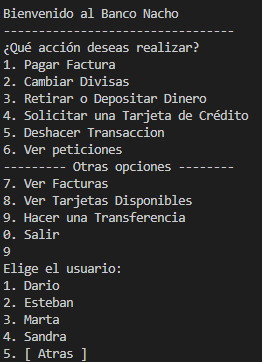
Aquí, el usuario tiene la opción de aceptar o rechazar cada una de las peticiones. En caso de que las rechace, no se hace ningún cambio en los fondos que tenga cada tarjeta. En caso de que acepte la petición, la transacción se deshace, es decir, el dinero que fué intercambiado en esta transacción se devuelve. En ambos casos (sin importar si rechaza o no) La transacción que fué escogida se convierte en una transacción no-retornable, es decir, si el cliente no podrá volver a crear una petición para que se le devuelva el dinero de esta transacción).

**hacerTransferencia (Opción número 6)**

**Manual de usuario**

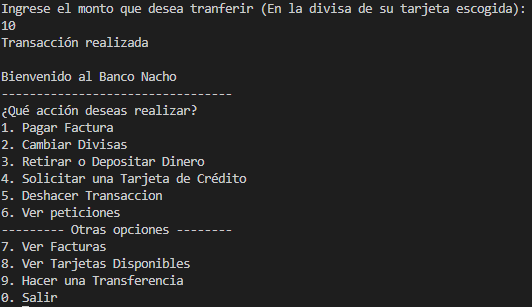
**Deshacer Transacción:**

Primero, en el menú principal, el usuario tendrá que crear alguna transacción. Esto se puede hacer mediante la opción 9. Después de elegir esta opción, el usuario tendrá que decidir desde cuál cliente hacer la transacción:



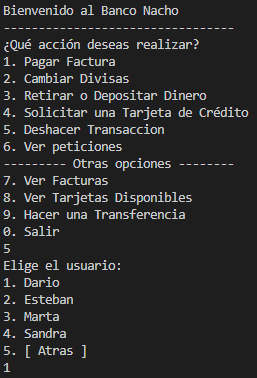
En este manual, escogeremos la opción 1, pero es posible hacer esto desde cualquier cliente.

Luego, el usuario tendrá que escoger la tarjeta desde la cual quiere hacer la transferencia, el cliente que recibirá la transacción, la tarjeta que recibirá la transacción, y el monto total de la transacción. Si todo va bien, al final el usuario recibirá el mensaje “Transacción realizada”:



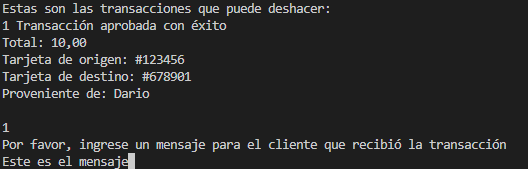
Ahora que el usuario ha hecho la transacción, podemos deshacerla:

Primero, necesitamos escoger opción “Deshacer Transacción” (opción 5). El programa nos dará la opción de escoger el cliente que queramos usar. Debemos escoger el cliente que hizo la transacción que queramos deshacer (En este ejemplo, Dario):



Luego, el programa nos da la opción de escoger mediante cuál criterio queremos filtrar las transacciones: Divisa (divisa de la tarjeta origen que se escogió al crear la transacción), Cliente que recibió la transacción, o tarjeta (La tarjeta de la cual salió la transacción. Podemos escoger cualquiera de estas tres opciones, siempre y cuando escojamos el criterio que corresponde con nuestra transacción.

Finalmente, nos aparece una lista de las transacciones que cumplen nuestro criterio. De esta lista escogemos la que queramos, y podemos escribir un mensaje para el cliente que recibió esta transacción:

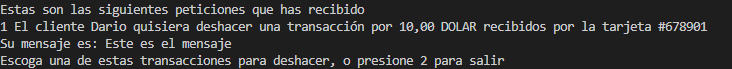


Al presionar enter, volvemos al menú principal. En este momento, hemos creado una “petición”, es decir, una transacción que queda pendiente y está esperando aprobación por parte del otro cliente para realizarse.

Para terminar esta funcionalidad, necesitamos escoger la opción 6 del menú principal (Ver peticiones).

Nuevamente, podemos escoger el cliente del cual vamos a ver las peticiones. Necesitamos escoger el cliente que recibió la transacción que queremos deshacer.

Al escoger el cliente, veremos una lista de peticiones que podemos deshacer.

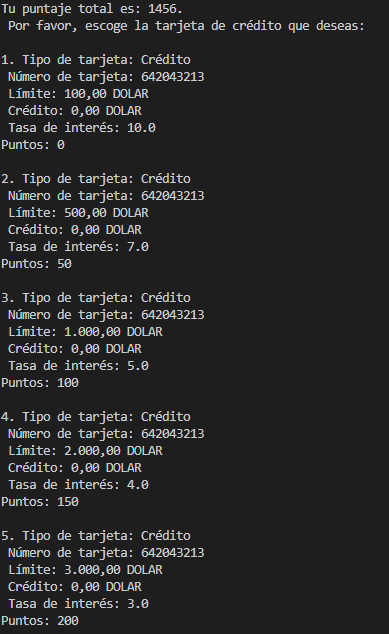


Aquí podemos escoger cualquiera de las peticiones, o podemos volver sin hacer nada.

Si escogemos una petición de esta lista, tendremos la opción de conceder o no conceder esta petición. En caso de que la concedamos, el dinero que fue transferido en esta transacción desaparecerá de la tarjeta que la recibió y volverá a la tarjeta de la cual salió. (Cosa que podemos verificar con la opción 8, ver tarjetas disponibles).

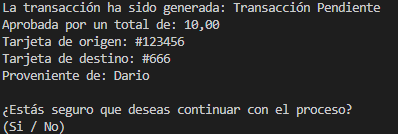
**Solicitar tarjeta de crédito**

Comenzamos escogiendo la opción 4 en el menú principal (Solicitar una tarjeta de crédito). Escogemos el cliente que queremos usar en esta operación, y luego escogemos la divisa que quisiéramos utilizar en esta operación. En la capa lógica, un algoritmo calcula un puntaje asociado al cliente que escogimos, y basándose en este puntaje, le da diferentes opciones para escoger. Podemos escoger una de estas opciones, y la tarjeta de crédito que escogimos es añadida a las tarjetas de crédito del cliente con el cual realizamos esta operación.



**Pagar factura**

Comenzamos escogiendo la opción 1 (Pagar factura). La primera decisión es con cual cliente hacer la operación. Luego, nos aparece una lista de facturas que corresponden al cliente que escogimos. Podemos escoger una de estas facturas que queremos pagar, y nos aparece una lista de tarjetas que podríamos utilizar para esta operación. Después de escoger la tarjeta que deseamos utilizar, tenemos que ingresar también la cantidad de dinero que queremos transferir. Finalmente, si el sistema aprueba la transacción el programa confirma si queremos completar la transacción:

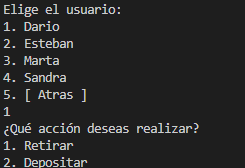


Si decidimos completar la transacción, se completa el pago y volvemos al menú principal. Podemos confirmar el efecto de esta funcionalidad mirando nuestras tarjetas o nuestras facturas.

**Retirar y depositar dinero**

Comenzando escogiendo la opción 3 en el menú principal (Retirar o Depositar Dinero).

Primero, debemos escoger el usuario que deseamos utilizar, como también la acción que queremos realizar (Retirar o depositar):



También debemos escoger la divisa con la cual queremos efectuar la operación, y luego la tarjeta correspondiente. El programa nos dará una lista de posibles canales mediante los cuales podemos efectuar la operación. (Los impuestos asociados a este canal se descontarán del total que se intenta retirar). Finalmente, el programa le pregunta al usuario por última vez si quiere continuar con el proceso, y si el usuario responde “si”, se finaliza la funcionalidad, cosa que se puede ver reflejada en el cambio en el estado tanto de la tarjeta como del canal utilizados.

**Cambiar Divisas**

Para usar esta funcionalidad, primero debemos usar la opción 2 en el menú principal (Cambiar Divisas). Luego, deberemos escoger el cliente con el cual realizar la operación, la divisa origen, y la divisa destino. Además, deberemos escoger las tarjetas de origen y destino, así como el canal que queremos utilizar, y el monto total que deseamos convertir. Si al final escogemos siempre opciones válidas, nos aparecerá esto al final:

