



# **Actividad 2 - Programa Banco Mexicano Parte 1**

## **Lenguajes de programación IV**

### **Ingeniería en Desarrollo de Software**

**Tutor: Aarón Iván Salazar Macías**

**Alumno: Jusi Ismael Linares Gutiérrez**

**Fecha: 04/07/2023**

## Índice

Introducción .....	3
Descripción .....	3
Justificación .....	3
Desarrollo: .....	4
Interfaz .....	4
Codificación.....	7
Conclusión .....	11

## Introducción

En el sector bancario, la atención a las necesidades de los clientes es primordial para brindarles un servicio eficiente y satisfactorio. El Banco Mexicano ha identificado la necesidad de desarrollar un programa que permita a sus clientes realizar depósitos, retiros y consultas de saldo de manera rápida y sencilla. Para cumplir con este objetivo, se requiere la creación de una base de datos que pueda manejar eficientemente esta información y que esté integrada con un programa en el lenguaje de programación Java 8.

En el caso de realizar un depósito, el programa permitirá al usuario ingresar la cantidad deseada y preguntará si se desea realizar otro depósito. En caso afirmativo, se mostrará nuevamente la pantalla de depósito para ingresar una nueva cantidad. En caso negativo, el programa regresará al menú principal.

Por otro lado, al seleccionar la opción de retiro, el programa solicitará al usuario la cantidad que desea retirar y luego volverá al menú principal.

## Descripción

El programa deberá contar con una pantalla principal que muestre un menú con las opciones mencionadas: Depósito, Retiro, Saldo y Salir. El usuario podrá seleccionar una de estas opciones ingresando su respuesta por teclado.

Si el usuario elige la opción de Depósito, el programa deberá solicitar la cantidad a depositar al usuario y preguntar si desea realizar otro depósito. En caso de seleccionar "Sí", se mostrará nuevamente la pantalla de Depósito para capturar los datos del siguiente depósito. Si se selecciona "No", se retornará al menú principal.

Por otro lado, si el usuario elige la opción de Retiro, el programa solicitará la cantidad a retirar al usuario y luego regresará al menú principal.

La implementación de este programa permitirá a los clientes del Banco Mexicano realizar operaciones bancarias de forma ágil y sencilla. Podrán hacer depósitos, retiros y consultar su saldo de manera conveniente, lo que contribuirá a mejorar su experiencia como usuarios del banco.

## Justificación

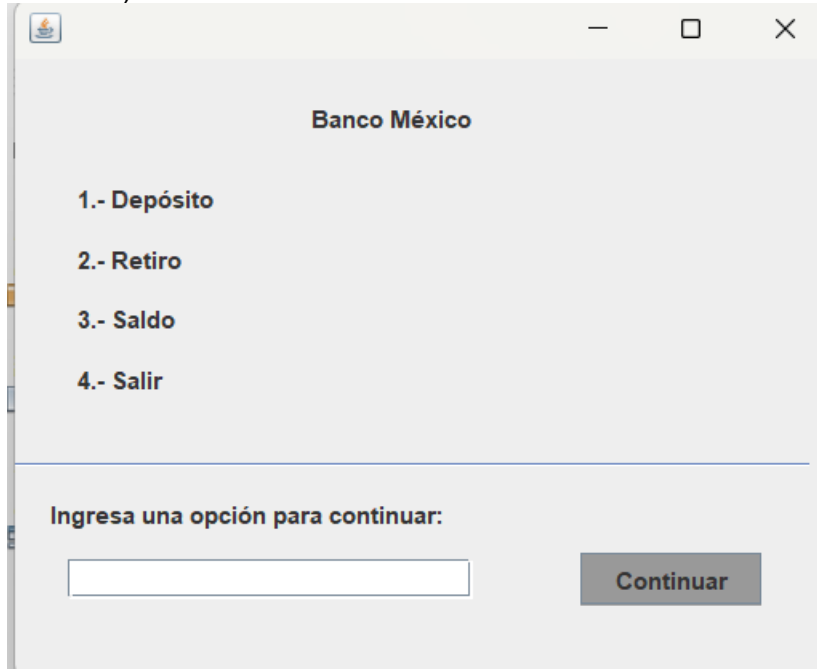
Java es un lenguaje de programación versátil y ampliamente utilizado en el desarrollo de aplicaciones empresariales. Proporciona una amplia gama de bibliotecas y herramientas que facilitan la conexión y manipulación de bases de datos. Además, Java ofrece una sintaxis clara y legible, lo que facilita el desarrollo de aplicaciones robustas y mantenibles.

Además, al tener un menú que ofrece opciones como depósito, retiro y consulta de saldo, se brinda una interfaz amigable y fácil de usar para los clientes. Esto les permite realizar transacciones bancarias de manera rápida y sencilla, sin la necesidad de acudir físicamente a una sucursal.

## Desarrollo:

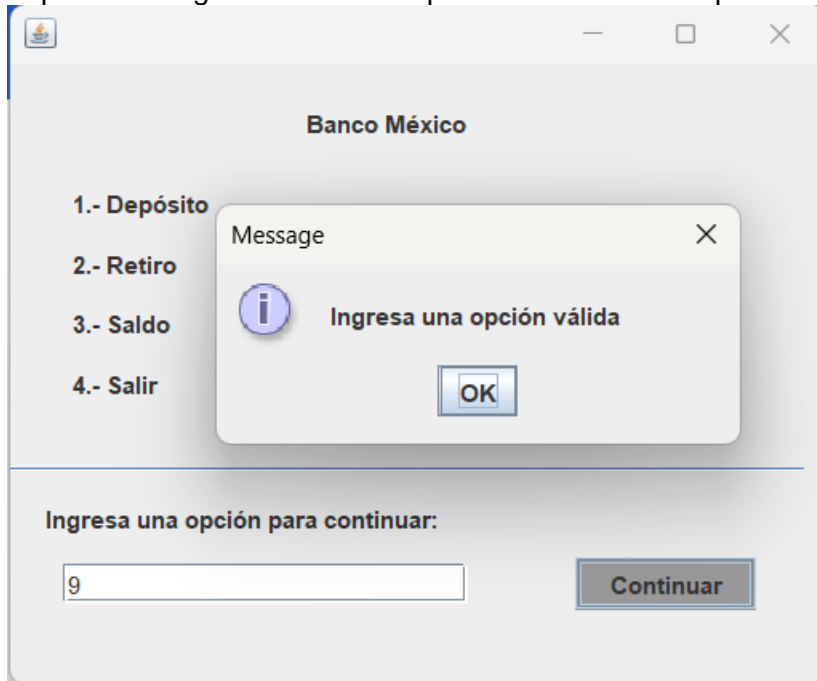
### Interfaz

En esta primera interfaz todo lo que debemos hacer es ingresar el número de la opción a la que queremos ingresar (Por ahora 3 y 4 no están programados ya que eso lo haremos en la siguiente actividad)



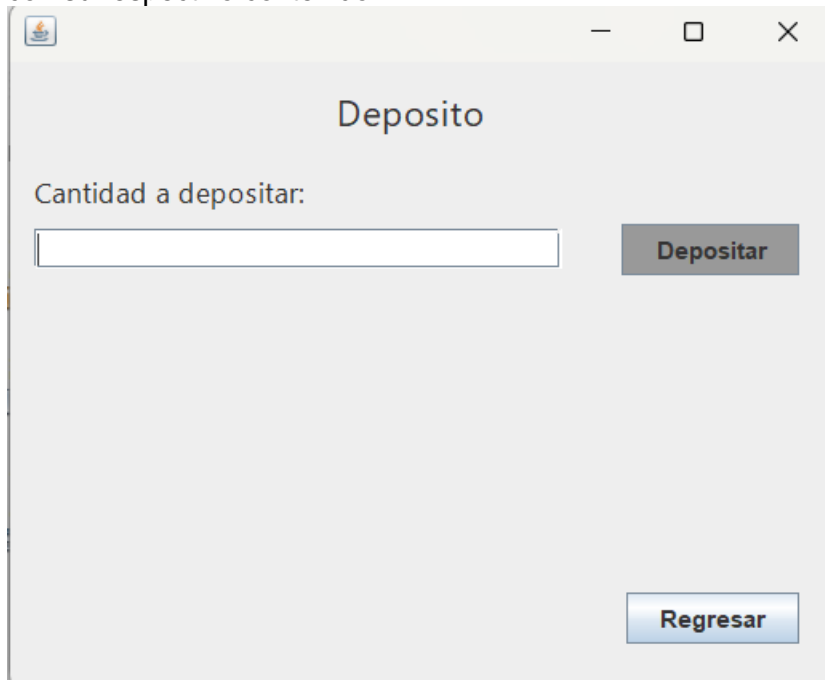
The screenshot shows a window titled "Banco México" with a standard Windows title bar (minimize, maximize, close buttons). Inside the window, there is a list of four options: "1.- Depósito", "2.- Retiro", "3.- Saldo", and "4.- Salir". Below this list, there is a label "Ingresa una opción para continuar:" followed by a text input field and a "Continuar" button.

Si ponemos algún otro número que no esté entre las opciones se nos mostrara un mensaje



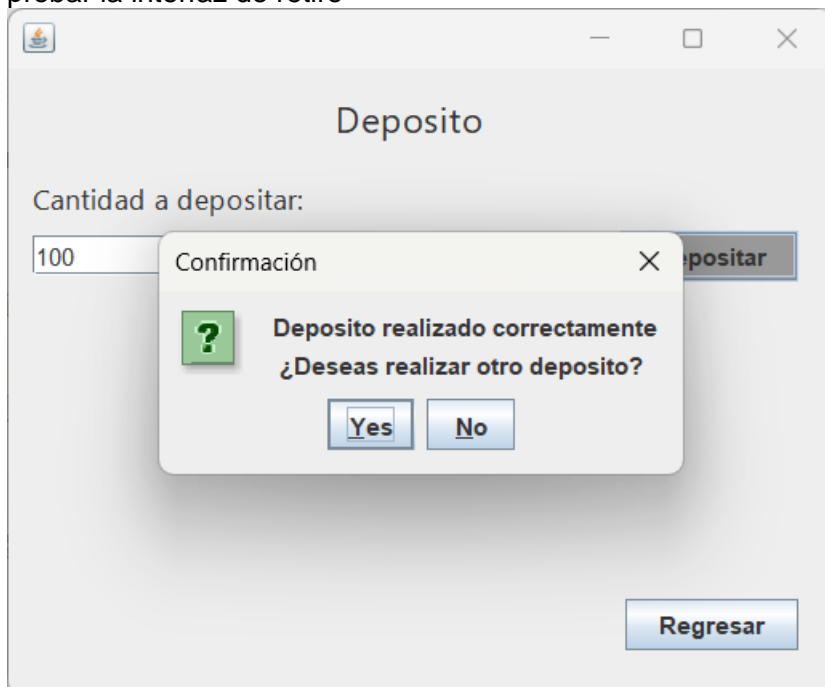
This screenshot shows the same "Banco México" window, but with an error message dialog box displayed over it. The dialog box is titled "Message" and contains an information icon (i) and the text "Ingresa una opción válida". There is an "OK" button at the bottom of the dialog. In the background window, the text input field now contains the number "9", and the "Continuar" button is still visible.

Una vez que ingresemos una opción valida como por ejemplo 1 se nos abrirá la siguiente interfaz con su respectivo contenido



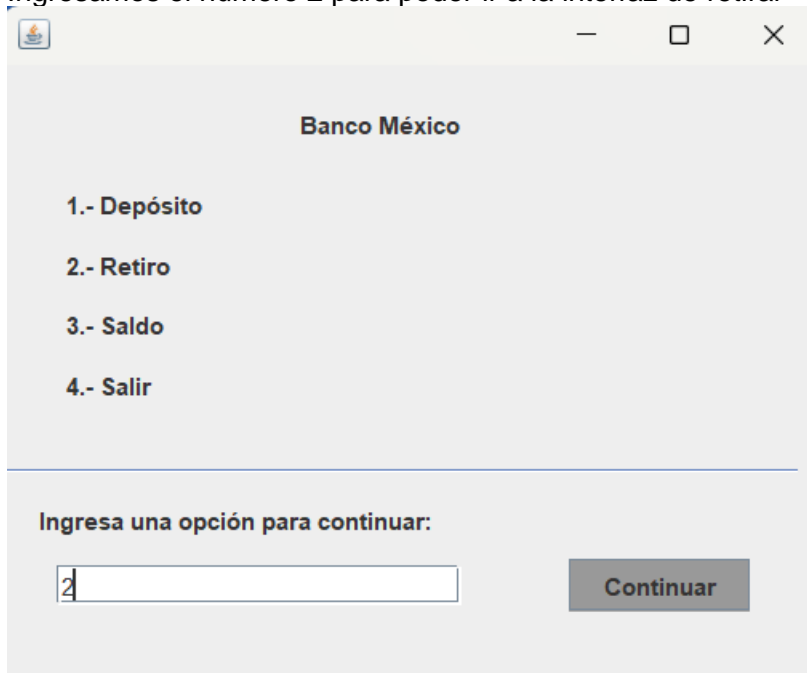
The screenshot shows a window titled "Deposito". Inside, there is a label "Cantidad a depositar:" followed by a text input field. To the right of the input field is a button labeled "Depositar". At the bottom right of the window is a button labeled "Regresar".

Ingresamos una cantidad a depositar y al dar clic en depositar se nos preguntara si queremos realizar otro deposito o no, si damos que sí nos quedaremos en la interfaz del depósito, en caso contrario simplemente nos regresara a la interfaz principal, daremos que No solo para continuar a probar la interfaz de retiro



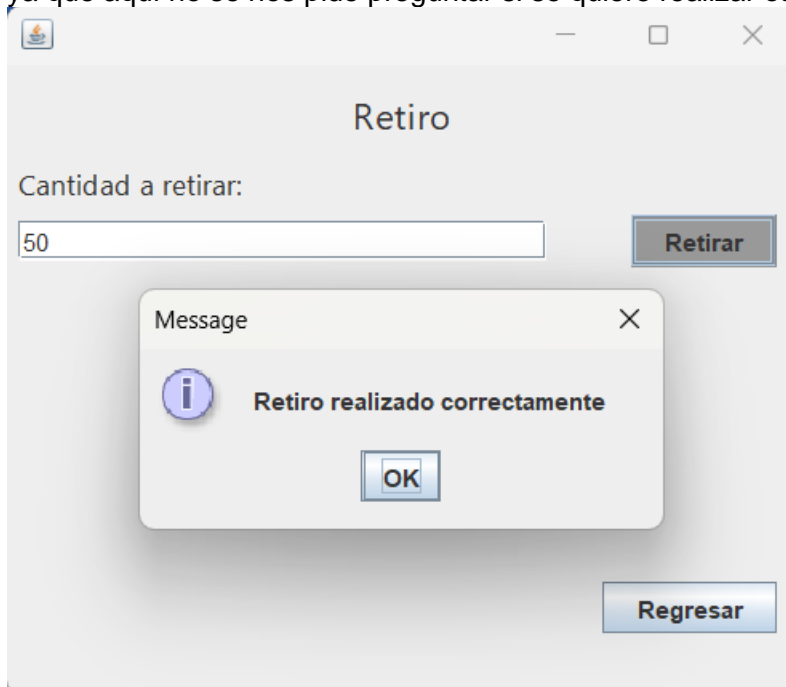
The screenshot shows the same "Deposito" window, but with a confirmation dialog box overlaid in the center. The dialog box has a title bar "Confirmación" and a close button. Inside the dialog, there is a green question mark icon, the text "Deposito realizado correctamente", and the question "¿Deseas realizar otro deposito?". Below the question are two buttons: "Yes" and "No". In the background window, the text input field now contains the number "100", and the "Depositar" button is visible.

Ingresamos el número 2 para poder ir a la interfaz de retirar



The screenshot shows a window titled 'Banco México' with a menu on the left containing four options: '1.- Depósito', '2.- Retiro', '3.- Saldo', and '4.- Salir'. Below the menu, there is a label 'Ingresa una opción para continuar:' followed by a text input field containing the number '2'. To the right of the input field is a button labeled 'Continuar'.

Ingresamos la cantidad a retirar y una vez realizado el retiro se nos mostrara un mensaje sobre que la operación se realizo correctamente, al dar clic en Ok se nos regresara a la interfaz principal ya que aquí no se nos pide preguntar si se quiere realizar otro retiro



The screenshot shows a window titled 'Retiro' (Withdrawal). It has a label 'Cantidad a retirar:' followed by a text input field containing the number '50'. To the right of the input field is a button labeled 'Retirar'. Below the input field, a message box is displayed with the title 'Message' and a close button 'X'. The message box contains an information icon (i) and the text 'Retiro realizado correctamente'. Below the message is an 'OK' button. At the bottom right of the 'Retiro' window is a button labeled 'Regresar'.

## Codificación

Al iniciar un nuevo proyecto en el archivo que nos abre ingresaremos este código

- 1.- Creamos una instancia de la clase MenuBancoMexico y lo asignamos a la variable frame
- 2.- Indicamos que al cerrar la ventana frame el programa deberá finalizar
- 3.- Vuelve visible la ventana frame

```
public class BancoMexicano {  
  
    public static void main(String[] args) {  
        1 MenuBancoMexico frame = new MenuBancoMexico();  
        2 frame.setDefaultCloseOperation(MenuBancoMexico.EXIT ON CLOSE);  
        3 frame.setVisible(true);  
    }  
}
```

Ahora en el JFrame del menú principal tendremos todo esto:

Con esto obtenemos lo escrito por el usuario en la caja de texto y lo guardamos en la variable opcionTxt

```
//Obtener la selección del usuario en el campo de texto  
String opcionTxt = OpcionMenu.getText();
```

Lo convertimos a int para que el switch case que utilizaremos después pueda leerlo sin problema

```
//Convertirlo a int  
int opcion = Integer.parseInt(opcionTxt);
```

Igual que hicimos anteriormente para abrir el menú principal abriremos la interfaz de deposito si el usuario ingreso el número 1 y la interfaz de retiro si el usuario ingreso el número 2

```
int opcion = Integer.parseInt(opcionTxt);  
  
//Switch case para cada una de las opciones, default para opciones no validas  
switch (opcion) {  
    case 1:  
        // Código a ejecutar para el caso 1 (Depósito)  
        //Abrir el JFrame de la interfaz de deposito  
        Deposito JFrameDeposito = new Deposito();  
        JFrameDeposito.setVisible(true);  
        break;  
    case 2:  
        // Código a ejecutar para el caso 2 (Retiro)  
        //Abrir el JFrame de la interfaz de retiro  
        Retirar JFrameRetirar = new Retirar();  
        JFrameRetirar.setVisible(true);  
        break;  
}
```

Esta parte aun no esta escrita debido a que esto se realizara en la siguiente actividad

```
case 3:  
    // Código a ejecutar para el caso 3  
    //Pendiente para la siguiente actividad  
    break;  
case 4:  
    // Código a ejecutar para el caso 4  
    //Pendiente para la siguiente actividad  
    break;
```

En el default solo haremos que se muestre un cuadro de texto que diga que se ingrese una opción válida

```
default:
    // Código a ejecutar en caso de que no se cumpla ningún caso anterior
    //double cantidad = VariablesGlobales.getCantidad(); Linea para probar que el s
    //javax.swing.JOptionPane.showMessageDialog(null, "La cantidad de dinero que ha
    //Cuadro de dialogo que indica que no se selecciono ninguna opción válida
    javax.swing.JOptionPane.showMessageDialog(null, "Ingresa una opción válida");
    break;
```

Ahora crearemos una nueva clase Java donde primero crearemos la variable cantidad del tipo double, tras esto creamos el método get que al llamarlo te da el valor que tenga la variable, por último creamos el método set donde al llamarlo debes ingresar un valor del tipo double para que se asigne a la variable cantidad y obtenga un nuevo valor

```
public class VariablesGlobales {
    //Declaración de la variable global cantidad tipo double
    private static double cantidad = 0;

    public static double getCantidad() {
        //Metodo para obtener el valor de la variable cantidad
        return cantidad;
    }

    public static void setCantidad(double transaccion) {
        //Metodo para asignarle un nuevo valor a la variable cantidad, ya sea deposito o retiro
        cantidad = transaccion;
    }
}
```

Ahora en el JFrame de deposito creamos una variable tipo double llamada cantidad y obtenemos el valor de la cantidad de dinero del usuario llamando al método getCantidad y asignando ese valor a nuestra nueva variable

```
//Obtenemos el valor de la variable cantidad que es el dinero con el que cuenta el usuario
double cantidad = VariablesGlobales.getCantidad();
```

Creamos una variable string y le asignamos lo escrito por el usuario en la textbox

```
//Obtenemos la cantidad que se va a depositar
String depositoTxt = depositoCantidad.getText();
```

La anterior variable string la convertiremos a double para que puedan realizarse operaciones con el

```
//La convertimos de string a double para que posteriormente se pueda sumar a la variable cantidad
double deposito = Double.parseDouble(depositoTxt);
```

Creamos una condición donde si la cantidad de deposito es mayor a 0 se ejecute el código correcto

```
//Validamos que haya ingresado una cantidad valida (no ceros ni números negativos
if (deposito > 0) {
```

A la variable cantidad que obtuvimos al inicio le sumaremos la cantidad a depositar por el usuario

```
//A la variable cantidad le sumamos el deposito que quiere realizar el usuario (cantidad = cantidad + deposito)
cantidad += deposito;
```

Ya que tenemos la nueva cantidad de dinero llamaremos al método set para cambiar el valor de la variable global cantidad por el del resultado de la suma

```
//Llamamos el metodo set para asignarle el nuevo valor obtenido de la suma anterior
VariablesGlobales.setCantidad(cantidad);
```



Con esta línea de código mostraremos un cuadro de confirmación donde se nos preguntara si queremos realizar otro deposito dando clic en "Yes" o en "No"

```
// Mostrar el cuadro de diálogo de confirmación
int continuarDeposito = JOptionPane.showConfirmDialog(this, "Deposito realizado correctamente \n "
+ "¿Deseas realizar otro deposito?", "Confirmación", JOptionPane.YES_NO_OPTION);
```

Si tu respuesta fue "Yes" el cuadro de confirmación simplemente se cierra para que puedas realizar otro deposito sin problema, en caso contrario se cerrara el JFrame de la interfaz de deposito para volver al menú principal

```
if (continuarDeposito == JOptionPane.YES_OPTION) {
    // Continuar realizando depositos
} else if (continuarDeposito == JOptionPane.NO_OPTION) {
    // Volver al menú principal cerrando la interfaz
    dispose();
}
```

Este else pertenece al if de más arriba donde se validaba que se ingresara una cantidad valida a depositar (no ceros ni negativos). En caso de no cumplir dicha condición se mostrará el mensaje donde pide ingresar una cantidad válida para depositar.

```
} else {
    javax.swing.JOptionPane.showMessageDialog(null, "Ingresa un número válido para depositar");
}
```

Por último, nos dirigimos al JFrame de la interfaz de retiro, iniciamos igual que en la interfaz de deposito donde obtenemos el valor de la variable global cantidad llamando al método get y asignando el valor a nuestra nueva variable local cantidad del tipo double

```
//Obtenemos el valor de la variable global cantidad
double cantidad = VariablesGlobales.getCantidad();
```

Creamos la variable retiroTxt donde se le asignara el valor de lo ingresado por el usuario en la textbox de la interfaz de retiro

```
//A la variable retiro Txt le asignamos la cantidad a retirar ingresada por el usuario
String retiroTxt = retiroCantidad.getText();
```

Convertimos la anterior variable del tipo string al tipo double

```
//Convertimos el anterior string a double
double retiro = Double.parseDouble(retiroTxt);
```

Verificamos que se ingreso una cantidad de retiro correcta (no ceros ni números negativos)

```
//Verificamos que dicha cantidad sea valida (No ceros ni números negativos)
if (retiro > 0) {
```

Si dicha condición se cumple se restara la cantidad a retirar ingresada por el usuario a la variable global cantidad, tras realizar la operación y asignar el resultado a nuestra variable cantidad, después llamaremos al método set donde asignaremos el resultado de la resta a la variable global cantidad y le mandaremos un mensaje al usuario informándole que la transacción se realizo correctamente.

```
if (retiro > 0) {  
    cantidad -= retiro;  
    VariablesGlobales.setCantidad(cantidad);  
    javax.swing.JOptionPane.showMessageDialog(null, "Retiro realizado correctamente");  
    dispose();  
} else {
```

Recordando el if anterior que valida la cantidad ingresada por el usuario, en caso de no cumplir con la condición se mandara un mensaje al usuario informándole que debe ingresar una cantidad de retiro valida

```
} else {  
    javax.swing.JOptionPane.showMessageDialog(null, "Ingresa una cantidad para retirar válida");  
}
```

Por último para los botones regresar de ambas interfaces (deposito y retiro) simplemente escribimos el código dispose(); al apretar cualquier botón de regresar, con esto se cierra la interfaz en la que se encuentre el usuario y lo regresa a la interfaz del menú

En la interfaz de deposito

```
private void RegresarActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    dispose();  
}
```

En la interfaz de retiro

```
private void RegresarActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    dispose();  
}
```

## Conclusión

En el campo laboral y en la vida cotidiana, la creación de programas que satisfacen las necesidades de los usuarios es de suma importancia. El desarrollo de software nos brinda la oportunidad de crear soluciones personalizadas que agilizan y automatizan tareas, mejorando la eficiencia y facilitando la interacción con los sistemas.

El uso del lenguaje de programación Java y la implementación de una base de datos permiten ofrecer a los clientes la posibilidad de realizar depósitos, retiros y consultas de saldo de manera rápida y conveniente. Esto no solo simplifica el proceso para los clientes, sino que también optimiza las operaciones del banco al reducir la necesidad de realizar estas transacciones de forma manual.

La creación de programas como este en el campo laboral o en la vida cotidiana nos permite automatizar tareas, mejorar la eficiencia y brindar una mejor experiencia al usuario. La capacidad de adaptar soluciones tecnológicas a las necesidades específicas de los usuarios es fundamental para optimizar procesos y simplificar tareas, lo que resulta en un beneficio tanto para las empresas como para los individuos en su vida diaria.