

DESCRIPCION DEL PROBLEMA

1) Una Corporación financiera le ha contratado para que le realice un sistema que permita administrar toda su red de cajeros automáticos. Los movimientos que debe el sistema controlar son:

Consultas

Depósitos

Retiros

El tipo de cuenta que se utilizará será solo la de **débito**.

De los usuarios el sistema debe contener: N° de identificación (9 dígitos), N° de cuenta (5 dígitos), N° de clave (4 dígitos), Nombre completo, Dirección y Saldo.

De los cajeros automáticos se necesita: Código Cajero (int), Dirección de su ubicación.

De las transacciones: N°, Fecha, Hora, Monto, saldo. Las consultas NO afectan el saldo.

Su programa deberá permitir que el usuario ingrese y realice la transacción que desee para lo cual usted debe crear en términos de programación una estructura de datos que permita controlar todas las transacciones e individualizar por su tipo (Consultas – Depósitos – Retiros). Tome en cuenta la forma real que se realizan y se registran las transacciones en los cajeros porque igual debe de administrarlas su sistema. Por ejemplo, para:

Un retiro registrar: Fecha, Hora de transacción, monto debitado, y saldo correspondiente (**comprobar saldo con retiro de acuerdo a los fondos que se tienen**).

Un depósito: Fecha, Hora de transacción, monto depositado, y nuevo saldo.

Una consulta: Fecha, hora transacción, Saldo.

Debe implementar una estructura dinámica para administrar los cajeros

Debe implementar una estructura dinámica para administrar los clientes o usuarios

Debe implementar una estructura dinámica para administrar las transacciones de los clientes en los respectivos cajeros.

Debe tener una opción que guarde toda la información en un archivo de nombre Entidad_financiera.txt

2) Utilizando el archivo Entidad_financiera.txt del punto anterior defina una opción en el Menú principal de su aplicación que deberá permitir realizar al operador o usuario de la financiera una búsqueda de cualquier cliente que en ese momento se desee revisar sus transacciones, por consiguiente, lo primero que deberá verificar es si el *número de clave* corresponde con algún registro del archivo de clientes que tiene la financiera. El conjunto de direcciones del que se dispone es de 1600 y va de 0 a 1599. Para esta búsqueda su sistema deberá implementar la opción de: **Búsqueda por dispersión** utilizando aritmética modular y resolviendo las posibles colisiones con direccionamiento por encadenamiento, para este algoritmo el sistema desplegará el lugar o dirección que ocupa el registro que se buscó. Una vez hecha la búsqueda dar la opción al usuario de la financiera para que muestre todas y cada una de las transacciones que tiene registrado ese determinado cliente.

3) Utilizando la información del punto primero del proyecto y como opción última del Menú principal llamada ORDENAR, su aplicación deberá en esta opción permitir los registros de todos los clientes sean ordenados con el método de ordenamiento **Shell** y **Quick Sort**.

Y permitir se muestre de nuevo la información, pero ahora ordenada dos veces una por cada método, además por cada uno debe presentar el número de comparaciones y el número de intercambios, y mostrar (de una forma llamativa) cuál fue el más eficiente para llevar a cabo el ordenamiento en esta estructura de datos.

Herramientas usadas

Computadora #1

Hardware

1. Procesador: Intel Core(TM) i3-6006 CPU 2.00GHZ
2. 4 gigas de RAM
3. Disco dura de 1 terabyte y 128gb de M.2 NVMe

Software

- Word 2016
- Eclipse java 2019 de 64 bits.

Computadora #2

Hardware

1. Procesador Intel Core i5 5200 CPU

2. 4 GB RAM

3. Disco duro de 1 tera

Software

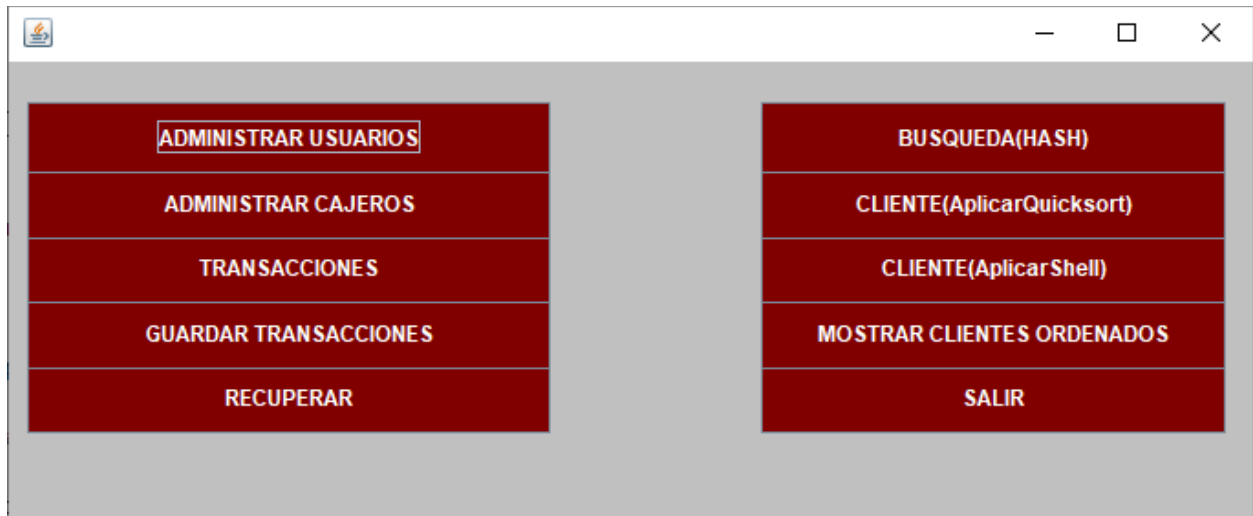
- Word 2016
- Eclipse java 2020 → Última versión para windows de 64 bits.

PROBLEMAS Y LIMITACIONES

1. Hubo dificultad de organización
2. Varios fallos de internet
3. Recuerde que para las transacciones se requiere el ingreso de los usuarios primeros

CORRIDO CON DATOS SUMINISTRADOS

Menú principal



Ingreso de Usuarios o Clientes:

Recuerde que se requiere de este ingreso de usuarios para poder usar la operación de transacción y las demás relacionadas a los clientes. También la administración del cliente es opcional ya que se puede ir directamente a las operaciones principales. Una vez ingresado los usuarios puede regresar al menú principal a proceder con las transacciones y lo demás.



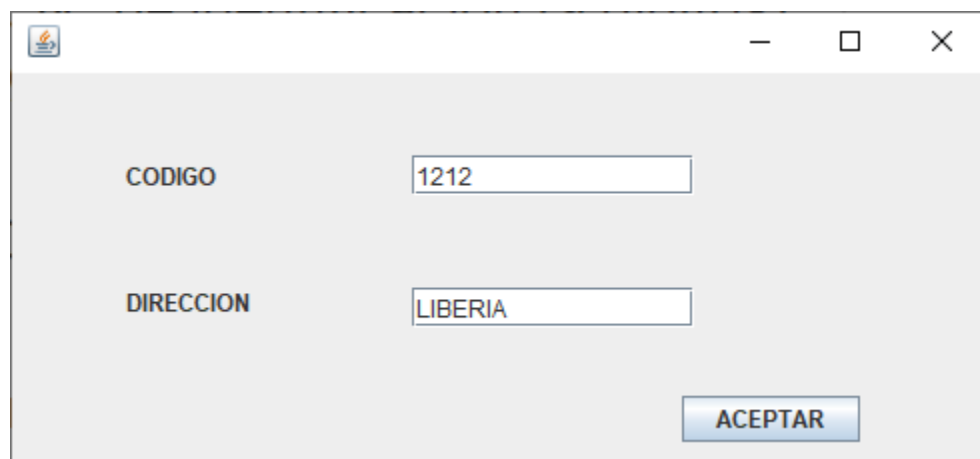


Formulario de datos de usuario con los siguientes campos:

Etiqueta	Valor
Nombre	JOSE
Numero De Identificacion	2323
Numero De Cuenta	545
CLAVE	5050
Direccion	LIBERIA
Saldo	5000

Botones de acción: ACEPTAR, CANCELAR

Ingreso de cajeros

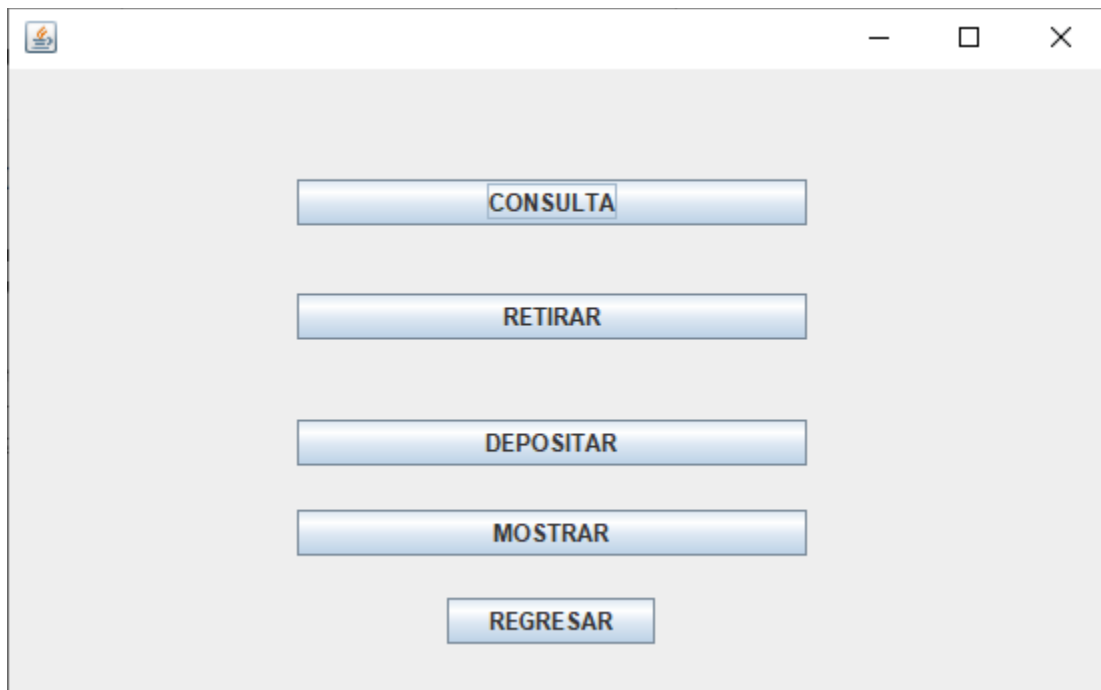


Formulario de ingreso de cajeros con los siguientes campos:

Etiqueta	Valor
CODIGO	1212
DIRECCION	LIBERIA

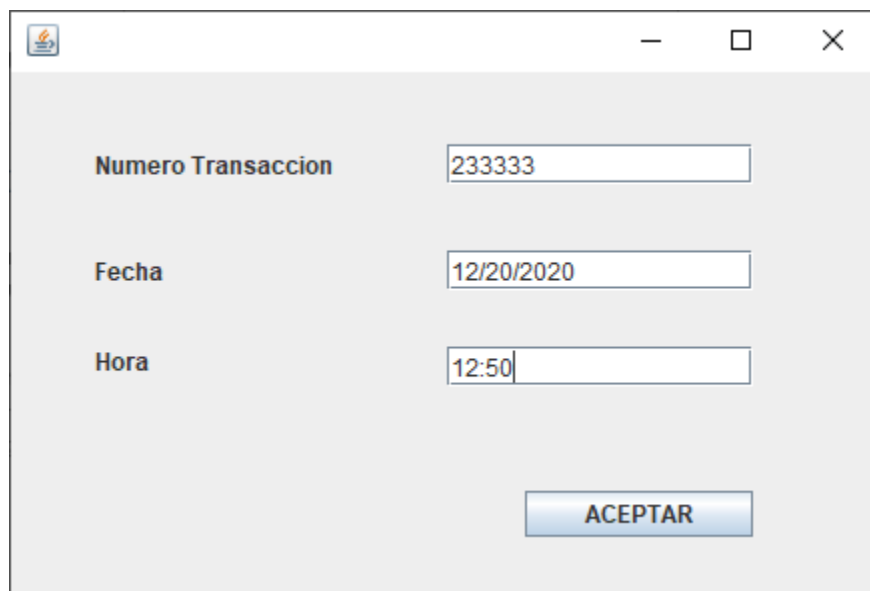
Botón de acción: ACEPTAR

Operaciones de transacciones

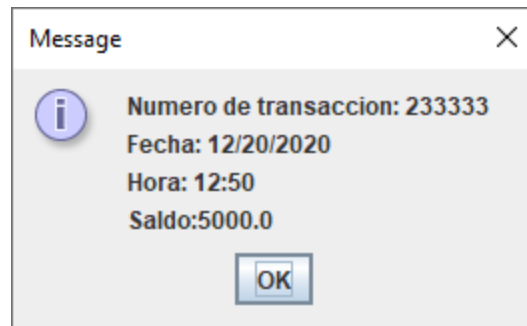


A screenshot of a software window titled "Operaciones de transacciones". The window has a standard Windows-style title bar with a small icon on the left and minimize, maximize, and close buttons on the right. The main area of the window is light gray and contains five blue buttons with white text, arranged vertically in the center. The buttons are labeled "CONSULTA", "RETIRAR", "DEPOSITAR", "MOSTRAR", and "REGRESAR".

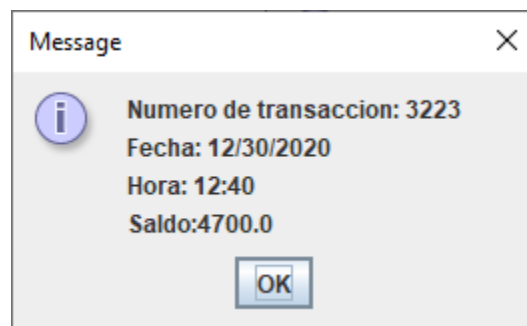
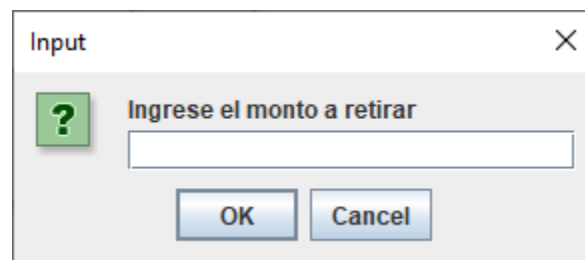
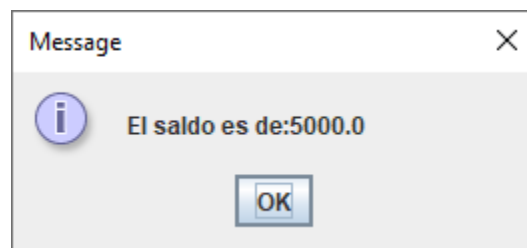
Consulta



A screenshot of a software window titled "Consulta". The window has a standard Windows-style title bar with a small icon on the left and minimize, maximize, and close buttons on the right. The main area is light gray and contains three labels on the left side, each followed by a text input field. The labels are "Numero Transaccion", "Fecha", and "Hora". The input fields contain the values "233333", "12/20/2020", and "12:50" respectively. At the bottom right of the window is a blue button with white text labeled "ACEPTAR".



Retirar




Deposito:

Input

 Digite la clave


OK Cancel

Message

 El saldo es de:4700.0


OK

Input

 Ingrese el monto a depositar

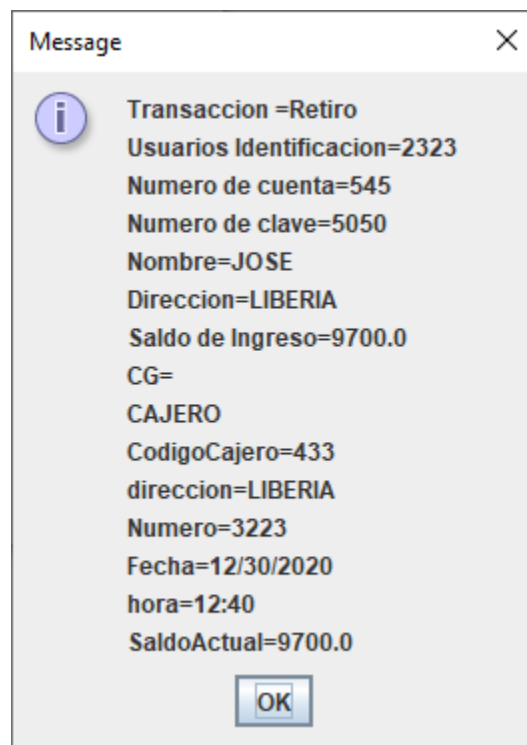
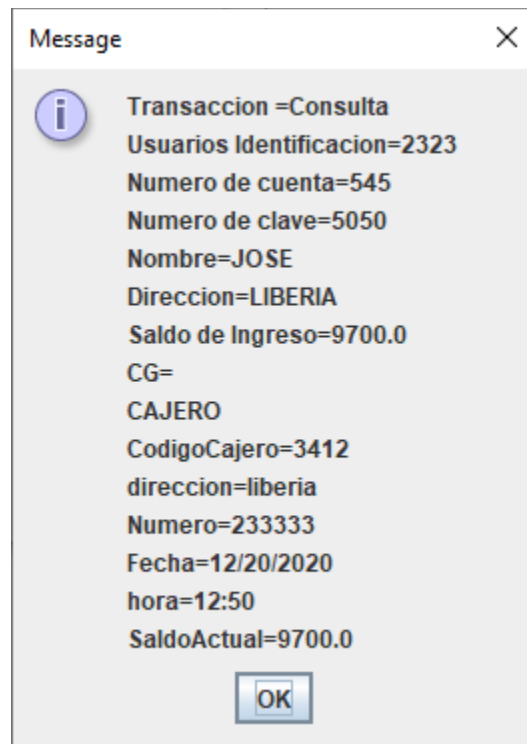
OK Cancel

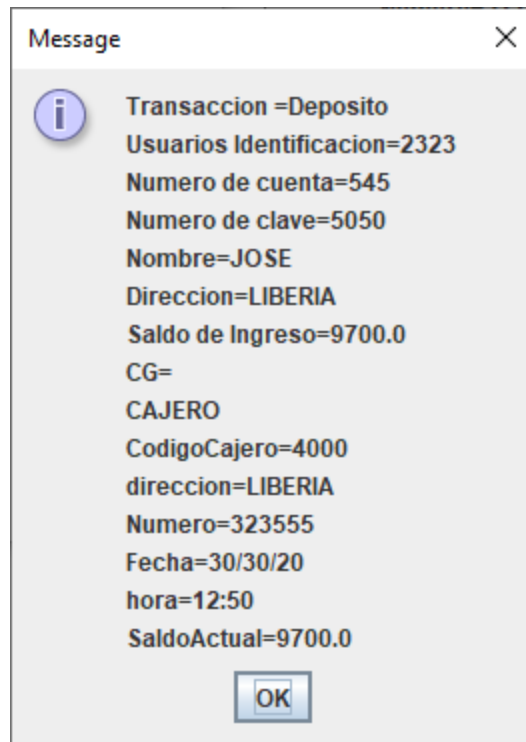
Message

 Numero de transaccion: 323555
Fecha: 30/30/20
Hora: 12:50
Saldo:9700.0

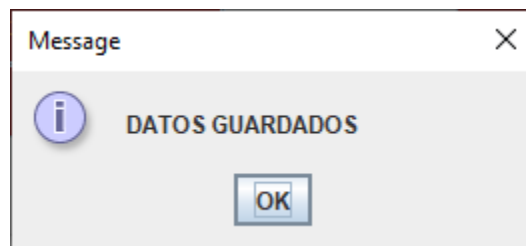
OK

Mostrar transacciones registradas

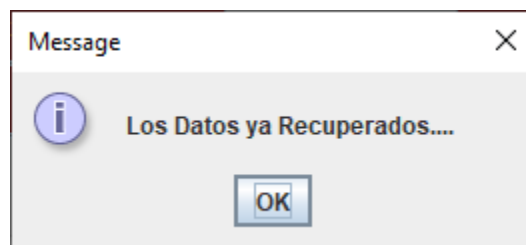




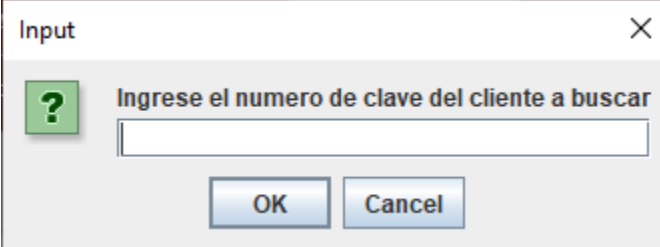
Guardar los datos en un archivo de texto



Recuerde recuperar los datos para después usarlos en la búsqueda por dispersión o hash:



Procedemos a hacer una búsqueda Hash por medio de la clave, permitiendo ver todas las transacciones que hizo ese cliente en específico y el registro que tiene en la.

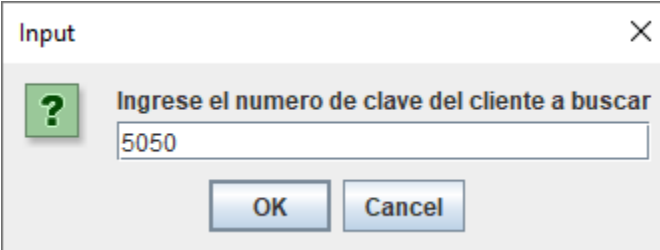


Input

?

Ingrese el numero de clave del cliente a buscar

OK Cancel

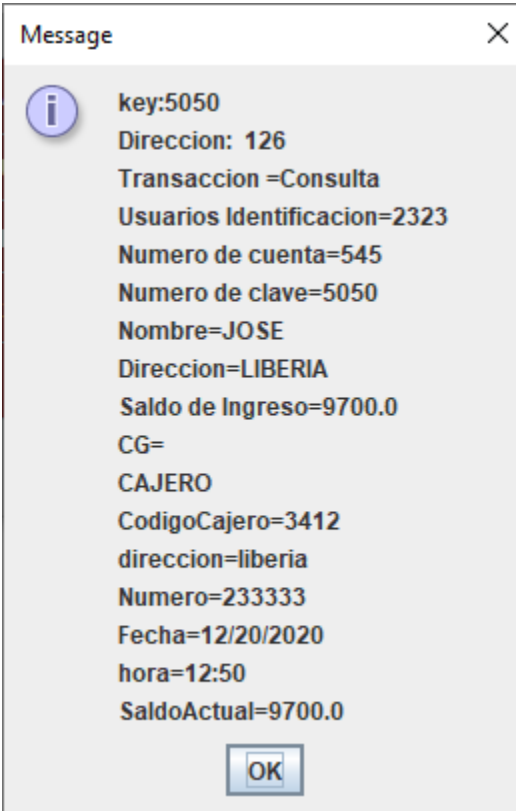


Input

?

Ingrese el numero de clave del cliente a buscar

OK Cancel

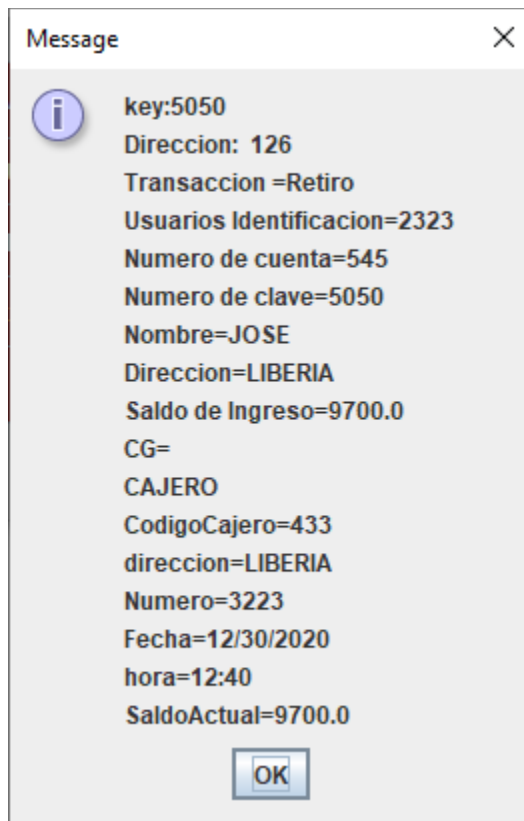


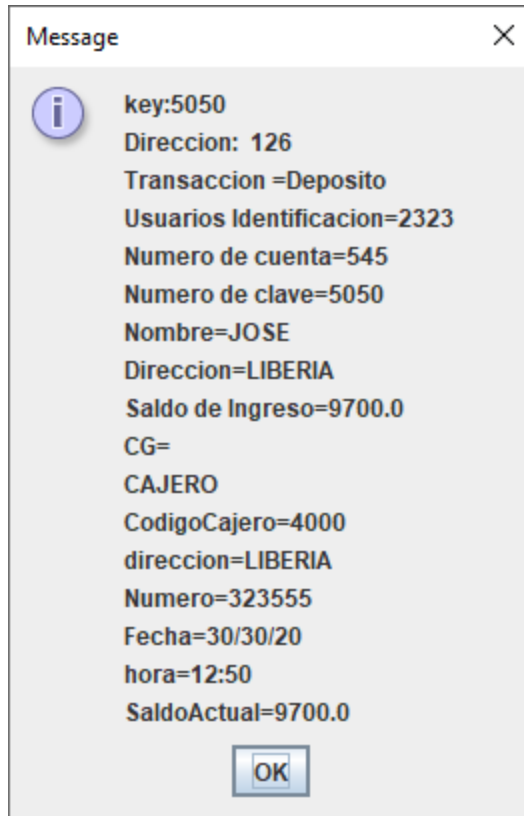
Message

i

key:5050
Direccion: 126
Transaccion =Consulta
Usuarios Identificacion=2323
Numero de cuenta=545
Numero de clave=5050
Nombre=JOSE
Direccion=LIBERIA
Saldo de Ingreso=9700.0
CG=
CAJERO
CodigoCajero=3412
direccion=liberia
Numero=233333
Fecha=12/20/2020
hora=12:50
SaldoActual=9700.0

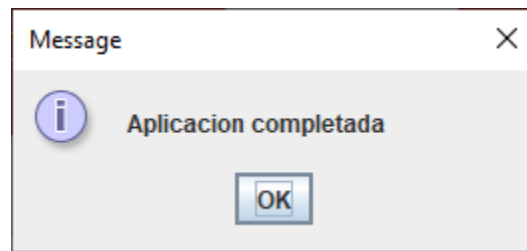
OK



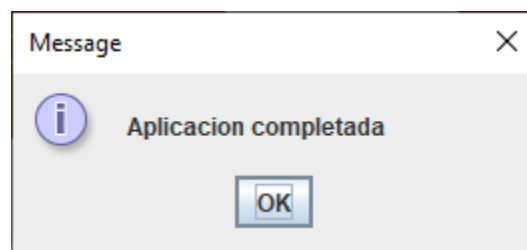


APLICACIONES DE ALGORITMOS

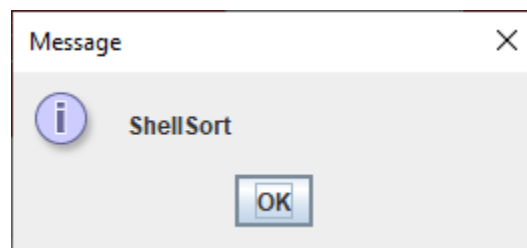
APLICAR QUICKSORT

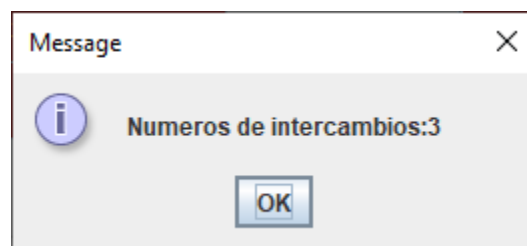
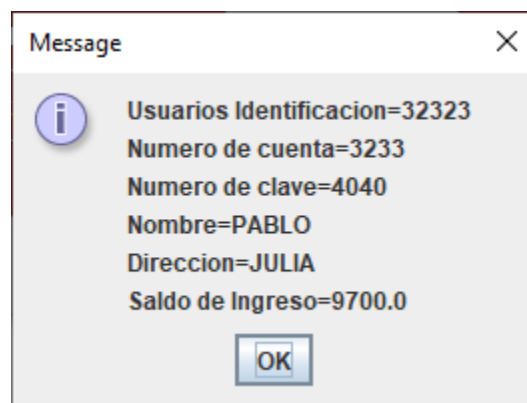
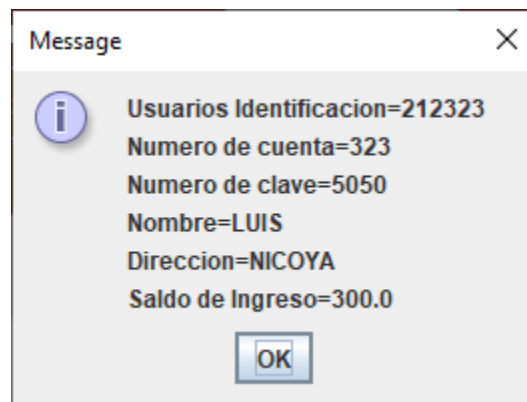
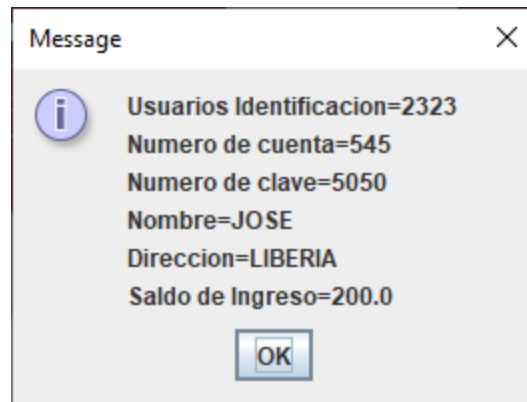


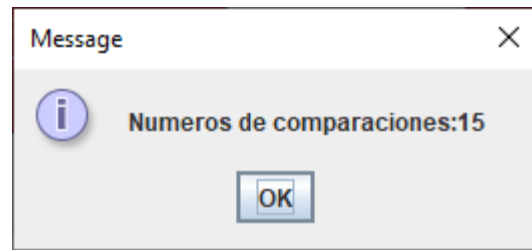
APLICAR SHELLSORT

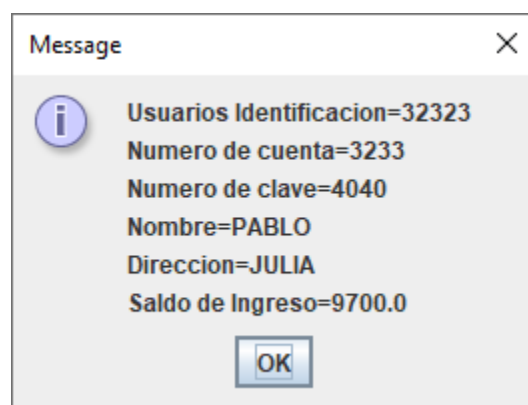
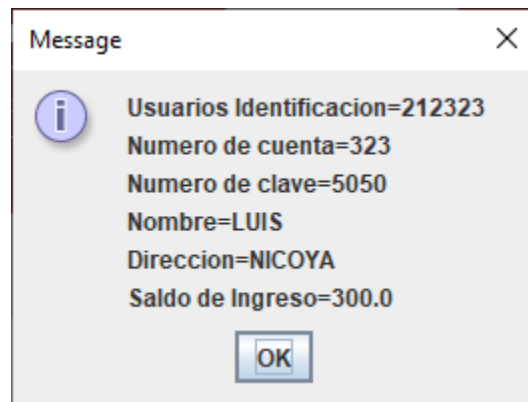
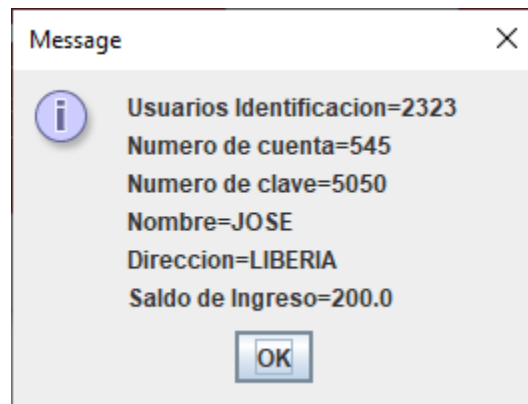
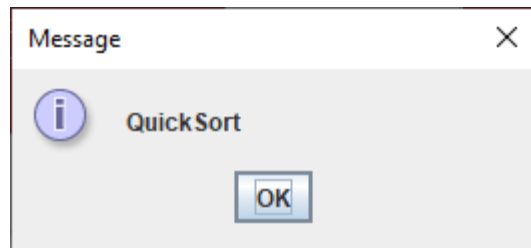


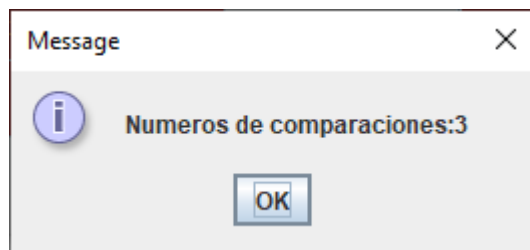
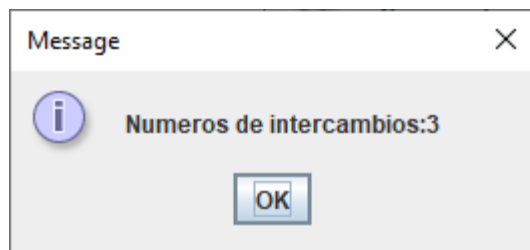
RECORRER Y COMPARAR











SOLUCIÓN DEL PROBLEMA

SOLUCION CAJEROS

```
import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.io.Serializable;
import java.awt.event.ActionEvent;

public class CAJEROSGUI implements Serializable{
    int CodigoCajero;
    String direccion;

    String Numero; //Numero de transaccion
    String Fecha; //Fecha de transaccion
    String hora; //Hora de transaccion
    int Monto; // Monto tanto de deposito como de retiro
    int SaldoT; //Saldo total

    private JFrame frame;
    private JTextField textField;
    private JTextField textField_1;

    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    CAJEROSGUI window = new CAJEROSGUI();
                    window.frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the application.
     */
}
```

```

    */
    public CAJEROSGUI() {

    }

    /**
     * Initialize the contents of the frame.
     */

    //Ingreso de datos
    public void Ingresar() {
        frame = new JFrame();
        frame.setBounds(100, 100, 504, 234);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.getContentPane().setLayout(null);

        JLabel lblNewLabel = new JLabel("CODIGO");
        lblNewLabel.setBounds(57, 44, 80, 14);
        frame.getContentPane().add(lblNewLabel);

        textField = new JTextField();
        textField.setBounds(200, 41, 141, 20);
        frame.getContentPane().add(textField);
        textField.setColumns(10);

        JLabel lblNewLabel_1 = new JLabel("DIRECCION");
        lblNewLabel_1.setBounds(57, 107, 92, 14);
        frame.getContentPane().add(lblNewLabel_1);

        textField_1 = new JTextField();
        textField_1.setBounds(200, 107, 141, 20);
        frame.getContentPane().add(textField_1);
        textField_1.setColumns(10);
        frame.setVisible(true);
        JButton btnNewButton = new JButton("ACEPTAR");
        btnNewButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {

                setCodigoCajero(Integer.parseInt(textField.getText()));
                setDireccion(textField_1.getText());

                frame.setVisible(false);
            }
        });
        btnNewButton.setBounds(335, 161, 89, 23);
        frame.getContentPane().add(btnNewButton);
    }

```

```

        public int getCodigoCajero() {
            return CodigoCajero;
        }

        public void setCodigoCajero(int codigoCajero) {
            CodigoCajero = codigoCajero;
        }

        public String getDireccion() {
            return direccion;
        }

        public void setDireccion(String direccion) {
            this.direccion = direccion;
        }

        @Override
        public String toString() {
            return "\nCAJERO \nCódigoCajero=" + CodigoCajero + "\ndireccion=" + direccion ;
        }

    }

import java.io.Serializable;

import javax.swing.JOptionPane;

public class LISTACAJEROS implements Serializable{

    //Nodos de almacenamiento en primero y ultimo
    NodoC primero;
    NodoC ultimo;

    int inter=0;
    int Com=0;

    public LISTACAJEROS() {
        primero=null;
        ultimo=null;

    }

    //El siguiente código inserta un nodo al inicio de la lista
    public void InsertarNodoInicio(CAJEROSGUI dato) {
        NodoC NodoNuevo=new NodoC();

```

```
NodoNuevo.dato=dato;
```

```
//1 y 2: Asignar el nodo y
```

```
//Poner en los datos * /
```

```
if(primeros==null) {
```

```
    primero=NodoNuevo;
```

```
    primero.siguiete=null;
```

```
    ultimo=primero;
```

```
}else {
```

```
    /// * 3. Haz el siguiete nodo nuevo como head * /
```

```
    ultimo.siguiete=NodoNuevo;
```

```
    NodoNuevo.siguiete=null;
```

```
/// * 4. Mueva la cabeza para apuntar al nuevo Nodo * /
```

```
    ultimo= NodoNuevo;
```

```
}
```

```
}
```

```
public void ImprimirLista(NodoC nodo) {
```

```
    if(nodo==null) {
```

```
        return;
```

```
    }
```

```
    JOptionPane.showMessageDialog(null, nodo.dato);
```

```
    ImprimirLista(nodo.siguiete);
```

```
}
```

```
public boolean BuscarCajero()
```

```
{
```

```
    NodoC current = primero; //Initialize current
```

```
    int DNI=Integer.parseInt(JOptionPane.showInputDialog("Ingresa el codigo del cajero a buscar"));
```

```
    while (current != null)
```

```
    {
```

```
        try {
```

```
            if (current.dato.CodigoCajero==DNI){
```

```
                JOptionPane.showMessageDialog(null, current.dato);
```

```

        //data found
    }
    }catch(Exception e) {}
    current = current.siguiente;

}
return false; //data not found
}

```

```

public void EliminarNodo() {
    NodoC actual=new NodoC();
    NodoC anterior=new NodoC();
    actual=primero;
    anterior=null;
    int DNI=Integer.parseInt(JOptionPane.showInputDialog("Ingrese el numero de telefono a remover"));
    while(actual!=null) {
        try {
            if(actual.dato.CodigoCajero==DNI) {
                if(actual==primero) {

                    actual=actual.siguiente;

                }
                else {

                    anterior.siguiente=actual.siguiente;

                }
            }

        }

    }

    }catch(Exception e) {}
    anterior=actual;
    actual=actual.siguiente;

}

}

```

```
}
```

```
public class NodoC {  
    CAJEROSGUI dato;  
  
    NodoC siguiente;  
  
    public NodoC() {}  
    public NodoC(CAJEROSGUI dato) {  
        this.dato=dato;  
        this.siguiente=null;  
    }  
}
```

```
}
```

```
TRANSACCION  
import java.io.File;  
import java.io.FileInputStream;  
import java.io.FileOutputStream;  
import java.io.IOException;  
import java.io.ObjectInputStream;  
import java.io.ObjectOutputStream;  
import java.io.Serializable;  
  
import javax.swing.JOptionPane;  
  
public class ListaTransaccion implements Serializable{  
  
    public NodoT[] table;           // storage for our nodes  
    public int countOfNodes;  
    int contador=0;  
    NodoT primero;  
    NodoT ultimo;  
  
    NodoT primero1;
```



```

NodoT ultimo1;
String nom1="";
String str="";
int sum=0;
String nom;

//Objetos paralelos temporales
TransaccionGUI T1=new TransaccionGUI(); //Objeto transaccion temporal
TransaccionGUI T2=new TransaccionGUI(); //Objeto transaccion temporal
TransaccionGUI T3=new TransaccionGUI(); //Objeto transaccion temporal
TransaccionGUI T4=new TransaccionGUI(); //Objeto transaccion temporal


//Modularizacion
public int getHash( String key ) {
    int hash = ( Math.abs( key.hashCode() ) ) % table.length;
    return hash;
}

public ListaTransaccion(int size )      {
    table = new NodoT[size];
    countOfNodes = 0;
}


public ListaTransaccion()      {
    //Direcciones de busqueda por dispersion
    this( 1601 );
    primero=null;
    ultimo=null;
    primero1=null;
    ultimo1=null;
}


// Este método permite ubicarme un objeto por medio de una clave
public TransaccionGUI get( String key ) {
    if ( key == null )
        throw new IllegalArgumentException( "Key is required." );
    int hash = getHash( key );

```

```

        if ( table[hash] == null ) {
            return null;
        }
        else {
            NodoT currentTable = table[hash];
            while ( currentTable != null ) {
                if ( currentTable.key.equals( key ) )
                    return currentTable.dato;
            }
            return null; // not found
        }
    }
}

```

//Metodo que permite buscar a un usuario por medio de una llave y mostralo
// con la respectiva direccion en la tabla
public TransaccionGUI Buscar(String key) {

```

        if ( key == null )
            throw new IllegalArgumentException( "Key is required." );
        int hash = getHash( key );
        if ( table[hash] == null ) {
            return null;
        }
        else {
            NodoT currentTable = table[hash];
            while ( currentTable != null ) {
                if ( currentTable.key.equals( key ) )
                    currentTable.key=currentTable.dato.U.Num_Clave;

```

```

                                JOptionPane.showMessageDialog(null, "key:" + currentTable.key
+ "\nDireccion: " + hash + "\n"+ currentTable.dato );
                                return currentTable.dato;

```

```

        }
        return null; // not found
    }
}

```

public TransaccionGUI BuscarTransaccion(String key) {

```

        if ( key == null )
            throw new IllegalArgumentException( "Key is required." );

```

```

        int hash = getHash( key );
        if ( table[hash] == null ) {
            return null;
        }
        else {

            for (int i = 0; i < table.length; i++) {

                if ( table[i] != null ) {

                    NodoT currentTable = table[i];
                    while ( currentTable != null ) {
                        if ( currentTable.key.equals( key ) )

                            JOptionPane.showMessageDialog(null,
"key:" + currentTable.key + "\nDireccion: " + hash + "\n" + currentTable.dato );
                            currentTable = currentTable.siguiente;
                        }

                    }

                }

            }

            return null; // not found
        }
    }

    public void put ( String key, TransaccionGUI value ) {
        if ( key == null )
            throw new IllegalArgumentException( "Key is required." );

        int hash = getHash( key );
        NodoT currentTable = table[hash]; // establece un puntero a la tabla que estamos
usando

        if ( currentTable != null ) {
            while ( currentTable.siguiente != null ) {
                currentTable = currentTable.siguiente;
                if ( currentTable.key.equals( key ) ) {
                    //overwriting entry that already exists
                    break;
                }
            }
        }
    }

```

```

    }
}

```

// Ahora estoy apuntando a una tabla vacía, un nodo con la misma clave o un nodo con un siguiente vacío

```

if ( table[hash] == null ) { // new Table
    currentTable = new NodoT();
    currentTable.key = key;
    currentTable.dato = value;
    table[hash] = currentTable;
    countOfNodes++;
}

```

```

else { // nuevo nodo agregado a la lista
    currentTable.siguiente = new NodoT();
    currentTable = currentTable.siguiente; //avanzar y actualizar el dato
    currentTable.key = key;
    currentTable.dato = value;
    countOfNodes++;
}

```

```

// for debug

```

```

}

```

//El siguiente código inserta un nodo al inicio de la lista
public void InsertarNodo(TransaccionGUI dato) {

```

    NodoT NodoNuevo=new NodoT();
    NodoNuevo.dato=dato;

```

```

    if(primeros==null) {
        primeros=NodoNuevo;
        primeros.siguiente=null;
        ultimo=primeros;

```

```

    }else {
        ultimo.siguiente=NodoNuevo;
        NodoNuevo.siguiente=null;
        ultimo= NodoNuevo;

```

```

    }

```

```
    contador++;
```

```
}
```

```
//El siguiente codigo inserta un nodo al inicio de la lista  
public void InsertarNodo1(TransaccionGUI dato) {
```

```
    NodoT NodoNuevo=new NodoT();  
    NodoNuevo.dato=dato;
```

```
    if(primer01==null) {  
        primero1=NodoNuevo;  
        primero1.siguiente=null;  
        ultimo1=primero1;
```

```
    }else {  
        ultimo1.siguiente=NodoNuevo;  
        NodoNuevo.siguiente=null;  
        ultimo1=        NodoNuevo;
```

```
}
```

```
    contador++;
```

```
}
```

```
public boolean Consulta()  
{  
    NodoT current= primero1; //Initialize current  
    NodoT anterior=new NodoT();  
    String DNI=JOptionPane.showInputDialog("Digite la clave");  
    while (current != null)  
    {  
        try {  
            if (current.dato.U.Num_Clave.equalsIgnoreCase(DNI)) {
```

```

        T1.U=current.dato.U;
        ConsultaDefinitivo();
        if(current.dato.Fecha.equalsIgnoreCase(null)) {
            if(current==primero1) {

                current=current.siguiente;

            }
            else {

                anterior.siguiente=current.siguiente;

            }

        }

        //data found
    }
    }catch(Exception e) {}
    current = current.siguiente;

}

return false; //data not found
}

public void ConsultaDefinitivo() {
    TransaccionGUI T=new TransaccionGUI();
    T.U=T1.U;
    T.IngresarConsulta();
T.Operation="Consulta";
    T3=T;
    InsertarNodo(T3);

    put ( T3.U.Num_Clave,T3 );

}

//Busque y retiro de algun usuario
public boolean Retiro()
{
    NodoT current = primero1; //Initialize current
    NodoT anterior=new NodoT();
    String DNI=JOptionPane.showInputDialog("Digite la clave");
    while (current != null)
    {
        try {
            if (current.dato.U.Num_Clave.equalsIgnoreCase(DNI)) {

```

```

JOptionPane.showMessageDialog(null, "El saldo es de:" + current.dato.U.Saldo);

T1.U=current.dato.U;
RetiroDefinitivo();
    if(current.dato.Fecha.equalsIgnoreCase(null)) {
        if(current==primero1) {

            current=current.siguiente;

        }
        else {

            anterior.siguiente=current.siguiente;

        }

    }

    //data found
}
}catch(Exception e) {}
current = current.siguiente;

}
return false; //data not found
}

```

```

public void RetiroDefinitivo() {
    TransaccionGUI T=new TransaccionGUI();
    T.U=T1.U;
    T.IngresarConsulta();;
    T.Operacion="Retiro";
    T.Monto=Integer.parseInt(JOptionPane.showInputDialog("Ingrese el monto a retirar"));
    int Calculo=(int) (T.U.Saldo-T.Monto);
    T.U.Saldo=Calculo;
    T2=T;

    InsertarNodo(T2);

    put ( T2.U.Num_Clave,T2 );
}

```

```

public boolean Deposito()
{
    NodoT current= primero1; //Initialize current
    NodoT anterior=new NodoT();
    String DNI=JOptionPane.showInputDialog("Digite la clave");
    while (current != null)
    {
        try {
            if (current.dato.U.Num_Clave.equalsIgnoreCase(DNI)) {

                JOptionPane.showMessageDialog(null, "El saldo es de:" + current.dato.U.Saldo);

                T1.U=current.dato.U;
                DepositoDefinitivo();
                if(current.dato.Fecha.equalsIgnoreCase(null)) {
                    if(current==primero1) {

                        current=current.siguiente;

                    }
                    else {

                        anterior.siguiente=current.siguiente;

                    }

                }

                //data found
            }
            }catch(Exception e) {}
            current = current.siguiente;

        }
        return false; //data not found
    }
}

```



```

        public void DepositoDefinitivo() {
            TransaccionGUI T=new TransaccionGUI();
            T.U=T1.U;
            T.IngresarConsulta();;
            T.Operation="Deposito";
            T.Monto=Integer.parseInt(JOptionPane.showInputDialog("Ingrese el monto a
depositar"));
            int Calculo=(int) (T.U.Saldo+T.Monto);
            T.U.Saldo=Calculo;
            T4=T;

```

```

            InsertarNodo(T4);
            put ( T4.U.Num_Clave,T4 );
        }
        public void ImprimirLista(NodoT nodo) {

            if(nodo==null) {
                return;

            }
            JOptionPane.showMessageDialog(null, nodo.dato);
            ImprimirLista(nodo.siguiente);
        }

```

```

        public void Guardar() throws IOException{

            NodoT gue=primero;
            File g=new File("Entidad_financiera.txt");
            ObjectOutputStream re = new ObjectOutputStream(new
FileOutputStream(g));

            while(gue!=null){
                re.writeObject(gue);
                gue = gue.siguiente;
            }
            JOptionPane.showMessageDialog(null, "DATOS GUARDADOS");
            re.close();

        }

```

```

        public void Recuperar() throws IOException, ClassNotFoundException{

```

```

        NodoT Leer;
        File Salud = new File("Entidad_financiera.txt");
        FileInputStream in;
        in = new FileInputStream(Salud);
        ObjectInputStream ver = new ObjectInputStream(in);
        if(primero==null){
            Leer = (NodoT) ver.readObject();
            primero=Leer;
            JOptionPane.showMessageDialog(null, "Datos Recuperados....");
        }else{
            JOptionPane.showMessageDialog(null, "Los Datos ya Recuperados....");
        }
        ver.close();
    }
}

```

```

}

```

```

import java.io.Serializable;

public class NodoT implements Serializable {
    TransaccionGUI dato;
    public String key;
    NodoT siguiente;

    public NodoT() {}
    public NodoT(TransaccionGUI dato) {
        this.dato=dato;
        this.siguiente=null;
    }

    public String getKey() {
        return key;
    }
    public void setKey(String key) {
        this.key = key;
    }
}

```

USUARIOS

```
import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JLabel;
import java.awt.Font;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.Color;
import java.awt.event.ActionListener;
import java.io.Serializable;
import java.awt.event.ActionEvent;

public class UsuariosGUI implements Serializable {
    String Num_Identificacion; // Identificacion del cliente
    String Num_Cuenta; // Numero de cuenta relacionada al usuario
    String Num_Clave; // Clave o contraseña relacionada al usuario
    String Nombre; // Nombre completo
    String Direccion; // Direccion del usuario
    double Saldo; // Saldo de ingreso y relacionado al usuario

    private JFrame frame;
    private JTextField textField;
    private JTextField textField_1;
    private JLabel lblNewLabel_2;
    private JTextField textField_2;
    private JLabel lblNewLabel_3;
    private JTextField textField_3;
    private JLabel lblNewLabel_4;
    private JTextField textField_4;
    private JLabel lblNewLabel_5;
    private JTextField textField_5;
    private JButton btnNewButton;
    private JButton btnNewButton_1;

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    UsuariosGUI window = new UsuariosGUI();
                    window.frame.setVisible(true);
                } catch (Exception e) {
                }
            }
        });
    }
}
```

```

                                e.printStackTrace();
                            }
                        }
                    });
}

public String getNum_Identificacion() {
    return Num_Identificacion;
}

public void setNum_Identificacion(String num_Identificacion) {
    Num_Identificacion = num_Identificacion;
}

public String getNum_Cuenta(){
    return Num_Cuenta;
}

public void setNum_Cuenta(String num_Cuenta) {
    Num_Cuenta = num_Cuenta;
}

public String getNum_Clave() {
    return Num_Clave;
}

public void setNum_Clave(String num_Clave){
    Num_Clave = num_Clave;
}

public String getNombre() {
    return Nombre;
}

public void setNombre(String nombre) {
    Nombre = nombre;
}

public double getSaldo() {
    return Saldo;
}

public void setSaldo(double saldo) {
    Saldo = saldo;
}

public String getDireccion() {
    return Direccion;
}

```

```

}

public void setDireccion(String direccion) {
    Direccion = direccion;
}

/**
 * Create the application.
 */
public UsuariosGUI() {
    initialize();
}

/**
 * Initialize the contents of the frame.
 */
private void initialize() {
    frame = new JFrame();
    frame.getContentPane().setForeground(new Color(192, 192, 192));
    frame.setBounds(100, 100, 640, 467);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.getContentPane().setLayout(null);

    JLabel lblNewLabel = new JLabel("Nombre");
    lblNewLabel.setFont(new Font("Arial Black", Font.PLAIN, 12));
    lblNewLabel.setBounds(40, 56, 117, 14);
    frame.getContentPane().add(lblNewLabel);

    textField = new JTextField();
    textField.setBounds(242, 44, 272, 35);
    frame.getContentPane().add(textField);
    textField.setColumns(10);

    JLabel lblNewLabel_1 = new JLabel("Numero De Identificacion");
    lblNewLabel_1.setFont(new Font("Arial Black", Font.PLAIN, 12));
    lblNewLabel_1.setBounds(40, 107, 210, 14);
    frame.getContentPane().add(lblNewLabel_1);

    textField_1 = new JTextField();
    textField_1.setBounds(242, 90, 272, 35);
    frame.getContentPane().add(textField_1);
    textField_1.setColumns(10);

    lblNewLabel_2 = new JLabel("Numero De Cuenta");
    lblNewLabel_2.setFont(new Font("Arial Black", Font.PLAIN, 12));
    lblNewLabel_2.setBounds(40, 155, 143, 14);
    frame.getContentPane().add(lblNewLabel_2);
}

```

```

textField_2 = new JTextField();
textField_2.setBounds(242, 145, 272, 37);
frame.getContentPane().add(textField_2);
textField_2.setColumns(10);

lblNewLabel_3 = new JLabel("CLAVE");
lblNewLabel_3.setFont(new Font("Arial Black", Font.PLAIN, 12));
lblNewLabel_3.setBounds(40, 206, 46, 14);
frame.getContentPane().add(lblNewLabel_3);

textField_3 = new JTextField();
textField_3.setBounds(242, 196, 272, 37);
frame.getContentPane().add(textField_3);
textField_3.setColumns(10);

lblNewLabel_4 = new JLabel("Direccion");
lblNewLabel_4.setFont(new Font("Arial Black", Font.PLAIN, 12));
lblNewLabel_4.setBounds(40, 261, 94, 14);
frame.getContentPane().add(lblNewLabel_4);

textField_4 = new JTextField();
textField_4.setBounds(242, 244, 272, 35);
frame.getContentPane().add(textField_4);
textField_4.setColumns(10);

lblNewLabel_5 = new JLabel("Saldo");
lblNewLabel_5.setFont(new Font("Arial Black", Font.PLAIN, 12));
lblNewLabel_5.setBounds(40, 315, 46, 14);
frame.getContentPane().add(lblNewLabel_5);

textField_5 = new JTextField();
textField_5.setBounds(242, 298, 272, 35);
frame.getContentPane().add(textField_5);
textField_5.setColumns(10);

btnNewButton = new JButton("ACEPTAR");
btnNewButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        setNombre(textField.getText());
        setNum_Identificacion(textField_1.getText());
        setNum_Cuenta(textField_2.getText());
        setNum_Clave(textField_3.getText());
        setDireccion(textField_4.getText());
        setSaldo(Integer.parseInt(textField_5.getText()));
    }
});
btnNewButton.setBackground(new Color(192, 192, 192));

```

```

        btnNewButton.setForeground(new Color(0, 0, 0));
        btnNewButton.setBounds(312, 371, 89, 23);
        frame.getContentPane().add(btnNewButton);

        btnNewButton_1= new JButton("CANCELAR");
        btnNewButton_1.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {

                frame.setVisible(false);

            }
        });
        btnNewButton_1.setBackground(new Color(192, 192, 192));
        btnNewButton_1.setBounds(451, 371, 104, 23);
        frame.getContentPane().add(btnNewButton_1);
    }
}

```

```
import java.io.Serializable;
```

```
public class NodoU implements Serializable {
    Usuarios dato; // Tipo de dato
```

```
    NodoU siguiente; // puntero
```

```
    public NodoU() {}
    public NodoU(Usuarios dato) {
        this.dato=dato;
        this.siguiente=null;
    }
}

```

```
}
```

OPERACIONES DE TRANSACCION

```
import java.awt.EventQueue;
```

```
import javax.swing.JFrame;
import javax.swing.JButton;
import java.awt.event.ActionListener;
```

```

import java.io.Serializable;
import java.awt.event.ActionEvent;

public class OPERACIONES implements Serializable {

    private JFrame frame;

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    OPERACIONES window = new OPERACIONES();
                    window.frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the application.
     */
    public OPERACIONES() {
        initialize();
    }

    /**
     * Initialize the contents of the frame.
     */
    private void initialize() {
        frame = new JFrame();
        frame.setBounds(100, 100, 565, 350);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.getContentPane().setLayout(null);
        frame.setVisible(true);
        JButton btnNewButton = new JButton("CONSULTA");
        btnNewButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
            }
        });
        btnNewButton.setBounds(144, 55, 255, 23);
        frame.getContentPane().add(btnNewButton);

        JButton btnNewButton_1 = new JButton("RETIRAR");
    }
}

```



```

        btnNewButton_1.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
            }
        });
        btnNewButton_1.setBounds(144, 112, 255, 23);
        frame.getContentPane().add(btnNewButton_1);

        JButton btnNewButton_2= new JButton("DEPOSITAR");
        btnNewButton_2.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
            }
        });
        btnNewButton_2.setBounds(144, 175, 255, 23);
        frame.getContentPane().add(btnNewButton_2);

        JButton btnNewButton_3= new JButton("REGRESAR");
        btnNewButton_3.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
            }
        });
        frame.setVisible(false);
    }
    });
    btnNewButton_3.setBounds(219, 264, 104, 23);
    frame.getContentPane().add(btnNewButton_3);

    JButton btnNewButton_4= new JButton("MOSTRAR");
    btnNewButton_4.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
        }
    });
    btnNewButton_4.setBounds(144, 220, 255, 23);
    frame.getContentPane().add(btnNewButton_4);
}
}
}

```

INTERFAZ PRINCIPAL

```

import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JButton;
import javax.swing.JPanel;

```

```

import java.awt.Color;
import java.awt.event.ActionListener;
import java.io.IOException;
import java.awt.event.ActionEvent;

public class PRINCIPAL{

    private JFrame frame; //Ventana principal
    private JFrame frame1; //Ventana de usuarios
    private JFrame frame2; //Ventana de cajeros
    private JFrame frame3; //Ventana de cajeros
    int exchanges= 0; //Intercambios metodo shell
    int comparisons= 0; //Intercambios comparaciones
    ListaUsuarios LU=new ListaUsuarios();
    LISTACAJEROS LC=new LISTACAJEROS();
    int contador;
    Usuarios US[]=new Usuarios[200];
    ListaTransaccion LT=new ListaTransaccion();
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    PRINCIPAL window = new PRINCIPAL();
                    window.frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the application.
     */
    public PRINCIPAL() {
        initialize();
    }

    /**
     * Initialize the contents of the frame.
     */
    private void initialize() {
        frame = new JFrame();
        frame.getContentPane().setBackground(new Color(192, 192, 192));
        frame.setBackground(new Color(0, 139, 139));
        frame.setBounds(100, 100, 696, 290);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.getContentPane().setLayout(null);
    }
}

```

```

JButton btnNewButton = new JButton("ADMINISTRAR USUARIOS");
btnNewButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        frame1 = new JFrame();
        frame1.setBounds(100, 100, 450, 300);
        frame1.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame1.getContentPane().setLayout(null);
        frame1.setVisible(true);
        JButton btnNewButton = new JButton("INGRESAR");
        btnNewButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                Usuarios U = new Usuarios();
                TransaccionGUI T = new TransaccionGUI();
                U.Ingresar();

                T.U = U;
                US[contador] = U;
                LU.InsertarNodoEmpleado(U);
                LT.InsertarNodo1(T);

                contador++;

            }
        });
        btnNewButton.setBounds(113, 27, 182, 23);
        frame1.getContentPane().add(btnNewButton);

        JButton btnNewButton_1 = new JButton("MOSTRAR");
        btnNewButton_1.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {

                LU.ImprimirLista(LU.primerO);

            }
        });
        btnNewButton_1.setBounds(113, 73, 182, 23);
        frame1.getContentPane().add(btnNewButton_1);

        JButton btnNewButton_2 = new JButton("BUSCAR");
        btnNewButton_2.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {

                LU.Buscar();

            }
        });
    }
});

```

```

        btnNewButton_2.setBounds(113, 122, 182, 23);
        frame1.getContentPane().add(btnNewButton_2);

        JButton btnNewButton_3= new JButton("ELIMINAR");
        btnNewButton_3.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {

                LU.EliminarNodo();
            }
        });
        btnNewButton_3.setBounds(113, 171, 182, 23);
        frame1.getContentPane().add(btnNewButton_3);
        frame1.setVisible(true);
        JButton btnNewButton_4= new JButton("REGRESAR");
        btnNewButton_4.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {

                frame1.setVisible(false);

            }
        });
        btnNewButton_4.setBounds(138, 219, 128, 31);
        frame1.getContentPane().add(btnNewButton_4);

    }

});
btnNewButton.setForeground(new Color(255, 255, 255));
btnNewButton.setBackground(new Color(128, 0, 0));
btnNewButton.setBounds(10, 22, 286, 39);
frame.getContentPane().add(btnNewButton);

JButton btnNewButton_1= new JButton("ADMINISTRAR CAJEROS");
btnNewButton_1.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        frame2= new JFrame();
        frame2.setBounds(100, 100, 468, 300);
        frame2.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame2.getContentPane().setLayout(null);
        frame2.setVisible(true);
        JButton btnNewButton = new JButton("REGISTRAR CAJEROS");
        btnNewButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                CAJEROSGUI CG=new CAJEROSGUI();
            }
        });
    }
});

```

```

        CG.Ingresar();
        LC.InsertarNodoInicio(CG);

    }

});
btnNewButton.setBounds(142, 26, 182, 23);
frame2.getContentPane().add(btnNewButton);

JButton btnNewButton_1= new JButton("BUSCAR");
btnNewButton_1.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        LC.BuscarCajero();

    }
});
btnNewButton_1.setBounds(142, 114, 182, 23);
frame2.getContentPane().add(btnNewButton_1);

JButton btnNewButton_2= new JButton("MOSTRAR");
btnNewButton_2.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        LC.ImprimirLista(LC.primerO);

    }
});
btnNewButton_2.setBounds(142, 71, 182, 23);
frame2.getContentPane().add(btnNewButton_2);

JButton btnNewButton_3= new JButton("ELIMINAR");
btnNewButton_3.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        LC.EliminarNodo();

    }
});
btnNewButton_3.setBounds(142, 167, 182, 23);
frame2.getContentPane().add(btnNewButton_3);

JButton btnNewButton_4= new JButton("REGRESAR");
btnNewButton_4.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        frame2.setVisible(false);

    }
});

```

```

        btnNewButton_4.setBounds(292, 227, 107, 23);
        frame2.getContentPane().add(btnNewButton_4);

    }

});
btnNewButton_1.setForeground(new Color(255, 255, 255));
btnNewButton_1.setBackground(new Color(128, 0, 0));
btnNewButton_1.setBounds(10, 58, 286, 39);
frame.getContentPane().add(btnNewButton_1);

JButton btnNewButton_2= new JButton("TRANSACCIONES");
btnNewButton_2.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        frame3= new JFrame();
        frame3.setBounds(100, 100, 565, 350);
        frame3.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame3.getContentPane().setLayout(null);
        frame3.setVisible(true);
        JButton btnNewButton= new JButton("CONSULTA");
        btnNewButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {

                LT.Consulta();

            }
        });
        btnNewButton.setBounds(144, 55, 255, 23);
        frame3.getContentPane().add(btnNewButton);

        JButton btnNewButton_1= new JButton("RETIRAR");
        btnNewButton_1.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {

                LT.Retiro();

            }
        });
        btnNewButton_1.setBounds(144, 112, 255, 23);
        frame3.getContentPane().add(btnNewButton_1);

        JButton btnNewButton_2= new JButton("DEPOSITAR");

```

```

        btnNewButton_2.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {

                LT.Deposito();

            }
        });
        btnNewButton_2.setBounds(144, 175, 255, 23);
        frame3.getContentPane().add(btnNewButton_2);

        JButton btnNewButton_3= new JButton("REGRESAR");
        btnNewButton_3.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {

                frame3.setVisible(false);

            }
        });
        btnNewButton_3.setBounds(219, 264, 104, 23);
        frame3.getContentPane().add(btnNewButton_3);

        JButton btnNewButton_4= new JButton("MOSTRAR");
        btnNewButton_4.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {

                LT.ImprimirLista(LT.primerero);

            }
        });
        btnNewButton_4.setBounds(144, 220, 255, 23);
        frame3.getContentPane().add(btnNewButton_4);

    }

});
btnNewButton_2.setBackground(new Color(128, 0, 0));
btnNewButton_2.setForeground(new Color(255, 255, 255));
btnNewButton_2.setBounds(10, 93, 286, 39);
frame.getContentPane().add(btnNewButton_2);

JButton btnNewButton_3= new JButton("GUARDAR TRANSACCIONES");
btnNewButton_3.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        try {

            LT.Guardar();

```

```

        } catch (IOException e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }

    }

});
btnNewButton_3.setForeground(new Color(255, 255, 255));
btnNewButton_3.setBackground(new Color(128, 0, 0));
btnNewButton_3.setBounds(10, 129, 286, 39);
frame.getContentPane().add(btnNewButton_3);

JButton btnNewButton_4= new JButton("RECUPERAR");
btnNewButton_4.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        try {
            LT.Recuperar();
        } catch (ClassNotFoundException | IOException e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }
    }
});
btnNewButton_4.setForeground(new Color(255, 255, 255));
btnNewButton_4.setBackground(new Color(128, 0, 0));
btnNewButton_4.setBounds(10, 164, 286, 39);
frame.getContentPane().add(btnNewButton_4);

JButton btnBusquedahash = new JButton("BUSQUEDA(HASH)");
btnBusquedahash.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String DNI=JOptionPane.showInputDialog("Ingrese el numero de clave
del cliente a buscar");
        LT.BuscarTransaccion(DNI);

    }
});
btnBusquedahash.setForeground(new Color(255, 255, 255));
btnBusquedahash.setBackground(new Color(128, 0, 0));
btnBusquedahash.setBounds(411, 22, 254, 39);
frame.getContentPane().add(btnBusquedahash);

JButton btnNewButton_5= new JButton("CLIENTE(AplicarQuicksort)");
btnNewButton_5.addActionListener(new ActionListener() {

```



```

        public void actionPerformed(ActionEvent e) {
            NodoU n = LU.primerO;
            while(n.siguiente != null) {
                n = n.siguiente;
            }
            LU.Quicksort(LU.primerO, n);

            JOptionPane.showMessageDialog(null, "Aplicacion completada");
        }
    });
    btnNewButton_5.setForeground(new Color(255, 255, 255));
    btnNewButton_5.setBackground(new Color(128, 0, 0));
    btnNewButton_5.setBounds(411, 58, 254, 39);
    frame.getContentPane().add(btnNewButton_5);

    JButton btnNewButton_6 = new JButton("CLIENTE(AplicarShell)");
    btnNewButton_6.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {

            ShellSort(US);

            JOptionPane.showMessageDialog(null, "Aplicacion completada");

        }
    });
    btnNewButton_6.setBackground(new Color(128, 0, 0));
    btnNewButton_6.setForeground(new Color(255, 255, 255));
    btnNewButton_6.setBounds(411, 93, 254, 39);
    frame.getContentPane().add(btnNewButton_6);

    JButton btnNewButton_7 = new JButton("MOSTRAR CLIENTES ORDENADOS");
    btnNewButton_7.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {

            JOptionPane.showMessageDialog(null, "ShellSort");
            printArray(US);

            JOptionPane.showMessageDialog(null, "QuickSort");
            LU.ImprimirLista1(LU.primerO);

            JOptionPane.showMessageDialog(null, "Numeros de intercambios:" +
LU.SwapCounter);

```

```

        JOptionPane.showMessageDialog(null,"Numeros de comparaciones:" +
LU.SwapCounter);

```

```

        }
    });
    btnNewButton_7.setForeground(new Color(255, 255, 255));
    btnNewButton_7.setBackground(new Color(128, 0, 0));
    btnNewButton_7.setBounds(411, 129, 254, 39);
    frame.getContentPane().add(btnNewButton_7);

    JButton btnNewButton_8= new JButton("SALIR");
    btnNewButton_8.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {

            System.exit(0);

        }
    });
    btnNewButton_8.setForeground(new Color(255, 255, 255));
    btnNewButton_8.setBackground(new Color(128, 0, 0));
    btnNewButton_8.setBounds(411, 164, 254, 39);
    frame.getContentPane().add(btnNewButton_8);
}

```

```

public int ShellSort(Usuarios arr[]) {

    int n = arr.length;

    // Start with a big gap, then reduce the gap
    for (int gap = n/contador; gap > 0; gap /= 2)
    {
        comparisons++;
        // Do a gapped insertion sort for this gap size.
        // The first gap elements a[0..gap-1] are already
        // in gapped order keep adding one more element
        // until the entire array is gap sorted
        for (int i = gap; i < n; i += 1)
        {

            // add a[i] to the elements that have been gap
            // sorted save a[i] in temp and make a hole at

```

```

        // position i
        Usuarios temp = arr[i];

        // shift earlier gap-sorted elements up until
        // the correct location for a[i] is found
        try {
            int j;
            for (j = i; j >= gap && arr[j - gap].Saldo > temp.Saldo; j -= gap)

                arr[j] = arr[j - gap];

            // put temp (the original a[i]) in its correct
            // location
            arr[j] = temp;
            exchanges++;
        } catch (Exception e1) {}
    }

    }
    return 0;
}

```

```

        public void printArray(Usuarios arr[])
        {
            int n = arr.length;
            for (int i=0; i<contador; ++i) {
                JOptionPane.showMessageDialog(null, arr[i] );
            }
        }
    }

```

```

        JOptionPane.showMessageDialog(null, "Numeros de intercambios:" + exchanges );

        JOptionPane.showMessageDialog(null, "Numeros de comparaciones:" + comparisons );
    }

}

```