

## ACTIVIDAD – SEMANA 07

Implementar una aplicación que permita controlar las facturas de una tienda comercial, para lo cual se debe crear tres clases en sus paquetes respectivos. La clase **Factura** (clase que controla los atributos privados de la factura), **ArregloFacturas** (clase que maneja el vector de tipo factura) y clase **frmVenta** (clase que interactúa con el usuario a través de la GUI).

Dentro de la clase **Factura** implementar:

- ❖ Atributos privados: nFactura (int), fecha (String), vendedor (String) y monto (double).
- ❖ Método constructor que inicialice los atributos.
- ❖ Métodos de acceso público get y set.

Dentro de clase **ArregloFacturas**:

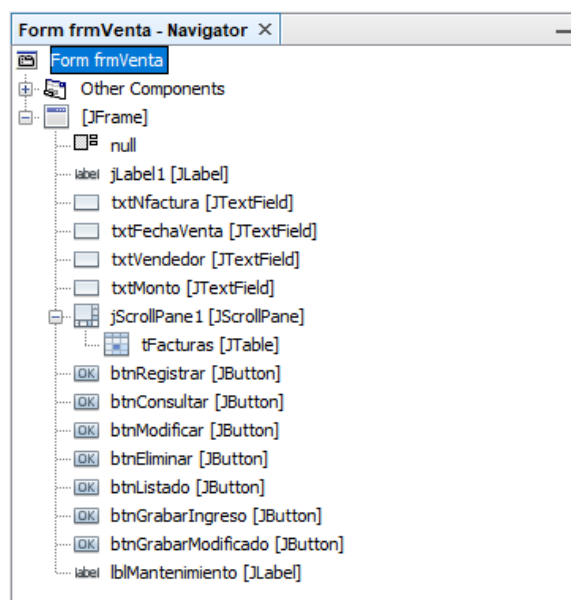
- ❖ Declarar como atributo privado el arreglo unidimensional **fact** de tipo **ArrayList** y el **índice**.
- ❖ Método **constructor** que inicialice el arreglo de **fact** de tipo **ArrayList**.
- ❖ Método **agregar**, que se encargará de registrar una factura en el **ArrayList**.
- ❖ Método **getTamaño**, que devuelve el tamaño del ArrayList.
- ❖ Método **obtener**, que devuelva todos los datos registrados en la Factura de acuerdo a la posición en el ArrayList.
- ❖ Método **buscar**, es el encargado de comparar si el **número de factura** ingresado existe en la matriz o no, dependiendo de eso deberá devolver el **objeto de la factura** encontrado, caso contrario devolver **null**. Aquí utilizar la estructura for de la siguiente forma:

```
for (Clase varReferencia: nombre del ArrayList){  
    return varReferencia;  
}
```

- ❖ Método **eliminar**, que dependiendo del tipo de objeto deberá eliminar lo del ArrayList.

En la clase **frmVenta**, diseñar las opciones de **mantenimiento** como **registrar** (para un nuevo registro de factura generando el número de factura en forma correlativa), **consultar** (que permite buscar una determinada factura), **modificar** (que permite modificar el nombre del vendedor y el monto registrado mas no el número y la fecha, ya que estos deben ser autogenerados) y **eliminar** (de acuerdo a un determinado objeto deberá eliminar un registro de empleado del ArrayList).

**SOLUCION:** Utilice la paleta de componentes y construya el siguiente diseño:



## REGISTRO DE VENTAS (FACTURACION)

Numero de Factura

1

Fecha Actual

2

Nombre del Vendedor

3

Monto a Registrar

4

Mantenimiento



5



6



7



8



9



10



11

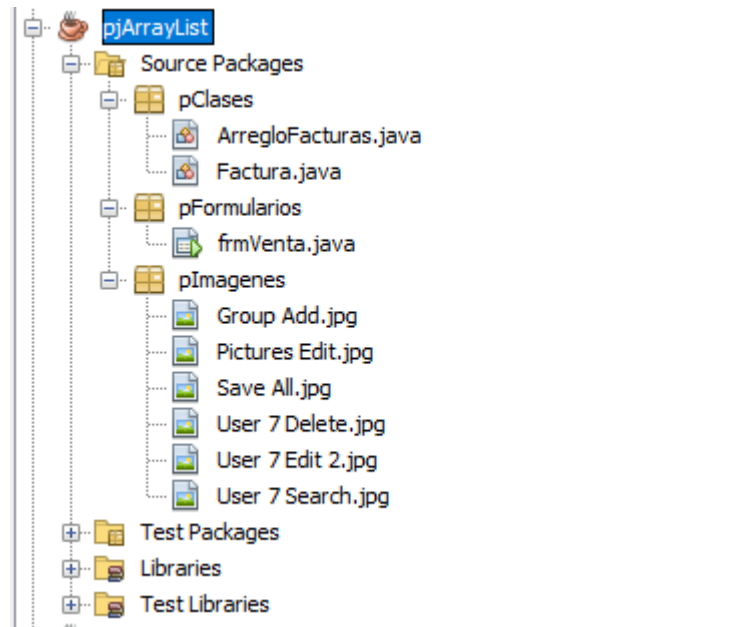
13

Nº Factura	Fecha de Venta	Vendedor	Monto
		12	

### Leyenda:

1. txtNfactura
2. txtFechaVenta
3. txtVendedor
4. txtMonto
5. btnGrabarIngreso
6. btnRegistrar
7. btnGrabarModificado
8. btnConsultar
9. btnModificar
10. btnEliminar
11. btnListado
12. tFacturas
13. lblMantenimiento

Observe la gráfica y su proyecto quedara de la siguiente forma que constara de dos clases y un formulario



A continuación, complete la información de la clase **Factura** con las siguientes instrucciones:

```
1  package pClases;
2
3  public class Factura {
4      private int nFactura;
5      private String fecha;
6      private String vendedor;
7      private double monto;
8
9      //Metodo constructor
10     public Factura(int nFactura,String fecha,String vendedor,double monto) {
11         this.nFactura=nFactura;
12         this.fecha=fecha;
13         this.vendedor=vendedor;
14         this.monto=monto;
15     }
16
17     //Metodos set
18     public void setNfactura(int nFactura) {
19         this.nFactura=nFactura;
20     }
21     public void setFecha(String fecha) {
22         this.fecha=fecha;
23     }
24     public void setVendedor(String vendedor) {
25         this.vendedor=vendedor;
26     }
27     public void setMonto(double monto) {
28         this.monto=monto;
29     }
30 }
```

```

30
31 //Metodos get
32 public int getNfactura() {
33     return nFactura;
34 }
35 public String getfecha() {
36     return fecha;
37 }
38 public String getVendedor() {
39     return vendedor;
40 }
41 public double getMonto() {
42     return monto;
43 }
44 }

```

A continuación, complete la información de la clase **ArregloFacturas** con las siguientes instrucciones:

```

1 package pClases;
2 import java.util.ArrayList;
3
4 public class ArregloFacturas {
5     private ArrayList <Factura> fact;
6     private int indice;
7
8     //Metodo constructor
9     public ArregloFacturas() {
10         fact=new ArrayList<Factura>();
11     }
12
13     //Metodo que devuelve el tamaño actual del vector
14     public int getTamaño() {
15         return fact.size();
16     }
17
18     //Metodo que permite agregar una factura al vector
19     public void agregar(Factura F) {
20         fact.add(F);
21     }
22
23     //Metodo que devuelve el objeto factura
24     public Factura obtener(int pos) {
25         return fact.get(pos);
26     }
27
28     //Metodo que busca una factura
29     public Factura buscar(int num) {
30         for (Factura f: fact)
31             if (f.getNfactura()==num)
32                 return f;
33         return null;
34     }
35
36     //Metodo que elimina una factura
37     public void eliminar(Factura x) {
38         fact.remove(x);
39     }
40 }

```

Váyase al editor de código y agregue los siguientes códigos y los siguientes métodos que se utilizarán para la solución del problema:

```
1 package pFormularios;
2 import java.util.GregorianCalendar;
3 import javax.swing.JOptionPane;
4 import javax.swing.table.TableColumn;
5 import pClases.ArregloFacturas;
6 import pClases.Factura;
7
8 public class frmVenta extends javax.swing.JFrame {
9
10     ArregloFacturas f=new ArregloFacturas();
11     int num=0;
12
13     public frmVenta() {
14         initComponents();
15         DefinirAnchos();
16         asignaFecha();
17         habilitaCajas(false);
18         btnGrabarIngreso.setVisible(false);
19         btnGrabarModificado.setVisible(false);
20     }
21
22     @SuppressWarnings("unchecked")
23     Generated Code
```

Instrucciones para el botón **Registrar**:

```
172 private void btnRegistrarActionPerformed(java.awt.event.ActionEvent evt) {
173
174     txtNfactura.setText(""+generaNumero());
175     asignaFecha();
176     txtVendedor.requestFocus();
177
178     habilitaCajas(true);
179     txtVendedor.setEditable(true);
180     txtMonto.setEditable(true);
181
182     txtVendedor.setText("");
183     txtMonto.setText("");
184
185     btnRegistrar.setVisible(false);
186     btnGrabarIngreso.setVisible(true);
187 }
188
```

Instrucciones para el botón **Consultar**:

```

188 private void btnConsultarActionPerformed(java.awt.event.ActionEvent evt) {
189     try{
190         limpiaCajas();
191         limpiaMatriz();
192         int buscoFactura=Integer.parseInt(JOptionPane.showInputDialog(null,"Ingrese un numero de Factura:"));
193         //objeto fact que busca el numero de factura en el ArrayList f
194         Factura fact=f.buscar(buscoFactura);
195         if (fact!=null){
196             tFacturas.setValueAt(fact.getNfactura(), 0, 0);
197             tFacturas.setValueAt(fact.getfecha(), 0, 1);
198             tFacturas.setValueAt(fact.getVendedor(), 0, 2);
199             tFacturas.setValueAt(fact.getMonto(), 0, 3);
200         }else
201             JOptionPane.showMessageDialog(null,"Factura NO encontrada","Confirmacion",JOptionPane.ERROR_MESSAGE);
202     } catch (Exception ex){
203         JOptionPane.showMessageDialog(null,"Error de Entrada de Datos","Confirmacion",JOptionPane.ERROR_MESSAGE);
204     }
205 }
206
207

```

Instrucciones para el botón **Modificar**:

```

209 private void btnModificarActionPerformed(java.awt.event.ActionEvent evt) {
210     try{
211         limpiaCajas();
212         limpiaMatriz();
213
214         btnModificar.setVisible(false);
215         btnGrabarModificado.setVisible(true);
216
217         int buscoFactura=Integer.parseInt(JOptionPane.showInputDialog(null,"Ingrese un numero de Factura:"));
218         //objeto fact que busca el numero de factura en el ArrayList f
219         Factura fact=f.buscar(buscoFactura);
220         if (fact!=null){
221             tFacturas.setValueAt(fact.getNfactura(), 0, 0);
222             tFacturas.setValueAt(fact.getfecha(), 0, 1);
223             tFacturas.setValueAt(fact.getVendedor(), 0, 2);
224             tFacturas.setValueAt(fact.getMonto(), 0, 3);
225
226             txtNfactura.setText(""+fact.getNfactura());
227             txtFechaVenta.setText(fact.getfecha());
228             txtVendedor.setText(fact.getVendedor());
229             txtMonto.setText(""+fact.getMonto());
230
231             habilitaCajas(true);
232             txtNfactura.setEditable(false);
233             txtFechaVenta.setEditable(false);
234         }else
235             JOptionPane.showMessageDialog(null,"Factura NO encontrada","Confirmacion",JOptionPane.ERROR_MESSAGE);
236     } catch (Exception ex){
237         JOptionPane.showMessageDialog(null,"Factura NO encontrada","Confirmacion",JOptionPane.ERROR_MESSAGE);
238         btnModificar.setVisible(true);
239         btnGrabarModificado.setVisible(false);
240     }
241 }
242

```

Instrucciones para el botón **Eliminar**:

```

241 private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
242     try{
243         int buscoFactura=Integer.parseInt(JOptionPane.showInputDialog(null,"Ingrese un numero de Factura a Eliminar:"));
244         //objeto fact que busca el numero de factura en el ArrayList f
245         Factura fact=f.buscar(buscoFactura);
246         if (fact!=null){
247             f.eliminar(fact);
248             JOptionPane.showMessageDialog(null,"Factura Eliminada Correctamente","Confirmacion",JOptionPane.INFORMATION_MESSAGE);
249             limpiaMatriz();
250             listar();
251         } else
252             JOptionPane.showMessageDialog(null,"NO existe el Numero de Factura ingresada","Confirmacion",JOptionPane.INFORMATION_MESSAGE);
253     } catch (Exception ex){
254         JOptionPane.showMessageDialog(null,"NO existe el Numero de Factura ingresada","Confirmacion",JOptionPane.INFORMATION_MESSAGE);
255     }
256 }
257

```

Instrucciones para el botón **Listado**:

```
258 private void btnListadoActionPerformed(java.awt.event.ActionEvent evt) {  
260     listar();  
261 }  
262
```

Instrucciones para el botón **Grabar Ingreso**:

```
262 private void btnGrabarIngresoActionPerformed(java.awt.event.ActionEvent evt) {  
264     try{  
265         habilitaCajas(false);  
266         btnRegistrar.setVisible(true);  
267         btnGrabarIngreso.setVisible(false);  
268  
269         Factura fact=new Factura (getNumFact(),getFecha(),getVendedor(),getMonto());  
270  
271         f.agregar(fact);  
272         listar();  
273         JOptionPane.showMessageDialog(null,"Factura ingresada correctamente","Confirmacion",JOptionPane.INFORMATION_MESSAGE);  
274     }catch(Exception ex){  
275         JOptionPane.showMessageDialog(null,"Error de Ingreso de Datos","Error",JOptionPane.ERROR_MESSAGE);  
276         num--;  
277     }  
278 }  
279
```

Instrucciones para el botón **Grabar Modificado**:

```
279 private void btnGrabarModificadoActionPerformed(java.awt.event.ActionEvent evt) {  
281     try{  
282         Factura fact=f.buscar(getNumFact());  
283         fact.setVendedor(getVendedor());  
284         fact.setMonto(getMonto());  
285         JOptionPane.showMessageDialog(null,"Factura Modificada Correctamente","Confirmacion",JOptionPane.INFORMATION_MESSAGE);  
286         listar();  
287     }catch(Exception ex){  
288         JOptionPane.showMessageDialog(null,"Ocurrio un error al intentar Grabar","Confirmacion",JOptionPane.INFORMATION_MESSAGE);  
289     }  
290     btnGrabarModificado.setVisible(false);  
291     btnModificar.setVisible(true);  
292 }
```

Métodos Adicionales:

```
293  
294 //Metodo que define el ancho de las columnas de la tabla  
295 void DefinirAnchos() {  
296     TableColumn columna;  
297     columna=tFacturas.getColumnModel().getColumn(0);  
298     columna.setPreferredWidth(30);  
299     columna=tFacturas.getColumnModel().getColumn(1);  
300     columna.setPreferredWidth(150);  
301     columna=tFacturas.getColumnModel().getColumn(2);  
302     columna.setPreferredWidth(150);  
303     columna=tFacturas.getColumnModel().getColumn(3);  
304     columna.setPreferredWidth(70);  
305     tFacturas.getTableHeader().setReorderingAllowed(false);  
306     tFacturas.getTableHeader().setResizingAllowed(false);  
307 }
```

```

308
309 //Metodo que bloque y desbloquea los controles JTextField
310 void habilitaCajas(boolean opcion){
311     txtNfactura.setEditable(opcion);
312     txtVendedor.setEditable(opcion);
313     txtMonto.setEditable(opcion);
314     txtFechaVenta.setEditable(opcion);
315
316 }

```

```

317
318 //Metodo que limpia los controles JTextField
319 void limpiaCajas(){
320     txtNfactura.setText("");
321     txtVendedor.setText("");
322     txtFechaVenta.setText("");
323     txtMonto.setText("");
324
325 }

```

```

326
327 //Metodo que limpiar el control tFacturas
328 void limpiaMatriz(){
329     for(int i=0;i<10;i++){
330         tFacturas.setValueAt("", i, 0);
331         tFacturas.setValueAt("", i, 1);
332         tFacturas.setValueAt("", i, 2);
333         tFacturas.setValueAt("", i, 3);
334     }
335 }

```

```

336
337 //Metodo que genera el numero de factura
338 public int generaNumero(){
339     num++;
340     return num;
341 }

```

```

342
343 //Metodos que capturan los valores ingresados por el usuario
344 public int getNumFact(){
345     return Integer.parseInt(txtNfactura.getText());
346 }
347 public String getFecha(){
348     return txtFechaVenta.getText();
349 }
350 public String getVendedor(){
351     return txtVendedor.getText();
352 }
353 public double getMonto(){
354     return Double.parseDouble(txtMonto.getText());
355 }

```



```

355
356 //Metodo que lista las facturas en el control tFacturas
357 void listar() {
358     if (f.getTamaño() > 0) {
359         for (int i = 0; i < f.getTamaño(); i++) {
360             Factura fact = f.obtener(i);
361             tFacturas.setValueAt(fact.getNfactura(), i, 0);
362             tFacturas.setValueAt(fact.getfecha(), i, 1);
363             tFacturas.setValueAt(fact.getVendedor(), i, 2);
364             tFacturas.setValueAt(fact.getMonto(), i, 3);
365         }
366     } else {
367         JOptionPane.showMessageDialog(this, "No hay facturas registradas", "Confirmacion", JOptionPane.INFORMATION_MESSAGE);
368         limpiaMatriz();
369     }
370 }
371


```

```

371
372 //Metodo que muestra la fecha actual en el control txtFechaVenta
373 void asignaFecha() {
374     GregorianCalendar cal = new GregorianCalendar();
375
376     txtFechaVenta.setText("" + cal.getTime());
377 }
378

```

Presione Shift+F6 y el aplicativo mostrará la siguiente ventana:


— □ ×

## REGISTRO DE VENTAS (FACTURACION)

**Numero de Factura**  

3

**Fecha Actual**  

Sep 29 00:12:38 PET 2021






**Nombre del Vendedor**  

Maximo

**Monto a Registrar**  

50

**Mantenimiento**

N° Factu...	Fecha de Venta	Vendedor	Monto
1	Wed Sep 29 00:11:57 PET 2021	Esther	500.0
2	Wed Sep 29 00:12:21 PET 2021	Marian	1000.0
3	Wed Sep 29 00:12:38 PET 2021	Maximo	50.0

\*Se requiere que analicen el código y expliquen al docente a cargo.