SEMANA 02

ENCAPSULAMIENTO

Clases administradoras con arreglo de objetos, clase ArrayList

CAPACIDAD DE PROCESO:

- Diseña clases administradoras utilizando la clase ArrayList
- Desarrolla métodos de procesos específicos con información del arreglo de objetos
- Desarrolla aplicaciones, con interface gráfica de usuarios, con objetos pertenecientes a clases administradoras

Clase administradora

Una clase administradora es aquella que tiene todo lo necesario para administrar información de cualquier tipo, considerando los atributos y funcionalidad necesarios para ello.

Arreglo de objetos

Un arreglo de objetos es aquel cuyos elementos que componen el arreglo son objetos. Es decir, cada casillero del arreglo es un objeto. Sin embargo, la posición de los casilleros siguen siendo enteros consecutivos a partir de 0.

Clase ArrayList

La clase **ArrayList** es una clase que administra un arreglo de objetos de cualquier tipo. No tiene restricciones de capacidad. Su tamaño se ajusta en forma dinámica utilizando eficientemente la memoria del computador.

- Tiene un Constructor por defecto que define un tamaño inicial de 10. Va creciendo de 10 en 10.

- Tiene varios constructores explícitos sin parámetros y con parámetros donde recibe el tamaño inicial del arreglo que usted quiera establecer.

- Los elementos dentro de un **ArrayList** son **Objetos** de cualquier tipo. Para un

uso adecuado se recomienda particularizarlo al objeto que se quiera administrar.

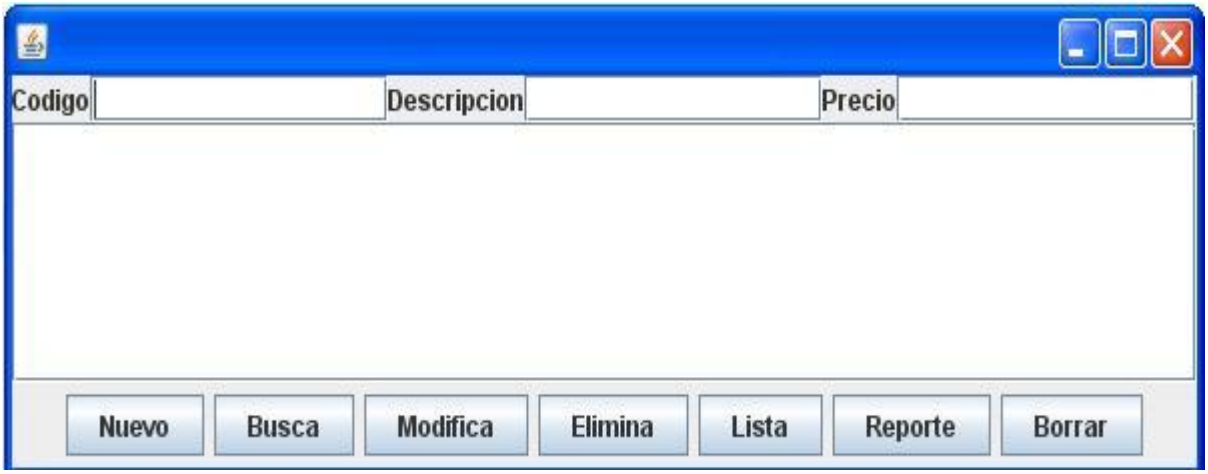
- La clase **ArrayList** forma parte del paquete **java.util**

Métodos de uso frecuente:

Método	Interpretación
int **size**()	Retorna el número de objetos guardados.
void **add**(Object obj)	Añade un objeto al final del arreglo, incrementando su tamaño actual en 1.
Object **get**(int index)	Devuelve el objeto almacenado a la posición index en el arreglo. **index** tiene que ser un entero entre **0** y **size()-1**.
void **set**(int index, Object obj)	reemplaza el objeto de la posición index por el objeto obj.
Object **remove**(int index)	Elimina el objeto a la posición index (index entre 0 y size()-1). Devuelve el objeto eliminado. Los objetos después de este objeto se trasladan a una posición anterior. El tamaño del ArrayList disminuye en 1.
int **indexOf**(Object obj)	Busca el objeto obj dentro del arreglo, si lo encuentra devuelve la posición donde lo ha encontrado. Si no lo encuentra, devuelve -1.

Ejemplo 1:

Diseñe una clase administradora para un arreglo de objetos de tipo Producto considerando la siguiente GUI:



Diseñe la clase **Producto** con constructor explícito y genere los métodos get-set:

```
public class Producto {
    // atributos privados
    private String codigo,
    descripcion; private double
    precio;

    // constructor
     public Producto(String codigo, String descripcion, double
        precio){ this.codigo=codigo;
        this.descripcion=descripcion;
        this.precio=precio;
    }
    // métodos get-set
}
```

Diseñe la clase **ArregloProductos**, que tenga como atributo privado un objeto de la clase **ArrayList** donde se guardarán los objetos y un constructor explícito sin parámetros.

```
public class ArregloProductos {
    // atributos
    // objeto  ArrayList particularizado para objetos de tipo
Producto
    private ArrayList <Producto> a ;

    // constructor explícito
    public ArregloProductos(){
        a = new ArrayList<Producto>();
    }
    // métodos de administración
    public int tamaño(){ return a.size(); }
    public void agrega(Producto p){
      a.add(p);
    }
```

```
    public Producto obtiene(int i){ // retorna un producto de la
posición i
        return a.get(i);
    }

    public void actualiza(Producto p, int i){
       // reemplaza un producto
       a.set(i, p);
    }
    // retorna la posición de un producto según
    su código. public int busca(String codigo){
        for(int i=0; i<tamaño(); i++){
            if(obtiene(i).getCodigo().equals(codigo)
            )
                return i;
        }
        return -1;  // no lo encontró
    }

    public void elimina(int p){// elimina el producto de la
       posición p a.remove(p);
    }

    public double mayorPrecio(){// retorna el mayor
        precio double m=a.get(0).getPrecio();
        for(Producto p: a){ // for each: por cada producto p en a
            if(p.getPrecio() > m)
                m =p.getPrecio();
        }
        return m;
    }

    public double menorPrecio(){// retorna el menor
        precio double m=a.get(0).getPrecio();
        for(Producto p: a){ // for each: por cada producto p en a
            if(p.getPrecio() < m)
                m =p.getPrecio();
        }
```

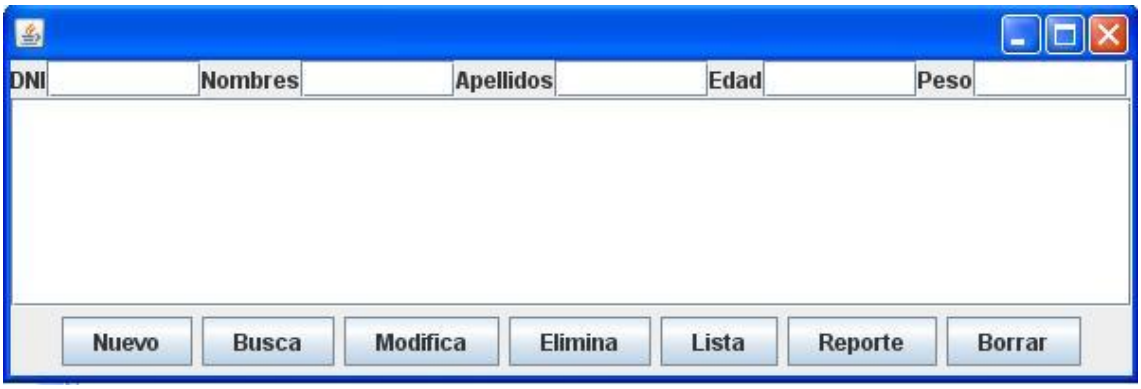
```
        return m;
    }
    public double precioPromedio(){// retorna el precio promedio
        double suma=0;
        for(Producto p: a){ // for each: por cada producto p en a
            suma += p.getPrecio();
        }
        return suma / tamaño();
    }
} // fin de la clase ArregloProductos
```

Ahora en la clase de la GUI considere como atributo un objeto de tipo ArregloProductos y programe la acción de los botones de acuerdo a lo siguiente:

- El botón Nuevo permite registrar un nuevo producto evitando que se repita el código
- El botón Busca ubica un producto según el código ingresado y muestra sus datos donde corresponda.
- El botón Modifica reemplaza los datos del producto ubicado con el botón Busca.
- El botón Elimina saca del arreglo el producto según el código ingresado.
- El botón Lista muestra una relación de todos los productos guardados en el arreglo.
- El botón Reporte muestra los precios mayor, menor y promedio.
- El botón Borrar limpia todo el contenido del GUI.

Ejemplo 2:

Diseñe una clase administradora con soporte para un arreglo de objetos tipo Persona considerando la siguiente GUI:



Diseñe la clase **Persona** con un constructor explícito:

```
public class Persona {
    // atributos privados
    private String dni, nombres,
    apellidos; private int edad;
    private double peso;
    // constructor explícito
    public Persona(String dni, String nombres, String apellidos,
                   int edad, double peso){
        this.dni=dni;
        this.nombres=nombres;
        this.apellidos=apellidos;
        this.edad=edad;
        this.peso=peso;
    }
    // métodos get-set
}
```

Diseñe la clase administradora **ArregloPersonas**, que tenga como atributo privado un objeto de la clase **ArrayList** particularizado para la clase **Persona**, donde se guardarán los objetos.

Desarrolle los siguientes métodos adicionales en la clase administradora:

a) Un método que retorne el peso mayor de todas las personas.
b) Un método que retorne el peso menor de todas las personas.
c) Un método que retorne el peso promedio de todas las personas.
d) Un método que retorne la cantidad de personas cuyo peso se encuentra en un rango dado como parámetros
e) Un método que retorne la cantidad de personas mayores de edad.
f) Un método que retorne la cantidad de personas menores de edad.

En la clase de la GUI considere como atributo un objeto de tipo ArregloPersonas y programe la acción de los botones de acuerdo a lo siguiente:

- El botón Nuevo permite registrar una nueva persona evitando que se repita el dni

- El botón Busca ubica una persona según el dni ingresado y muestra sus datos donde corresponda.
- El botón Modifica reemplaza los datos de la persona ubicada con el botón Busca.
- El botón Elimina saca del arreglo la persona según el dni ingresado.
- El botón Lista muestra una relación de todas las personas guardadas en el arreglo.
- El botón Reporte muestra el resultado de todos los métodos adicionales.
- El botón Borrar limpia todo el contenido del GUI.

Ejemplo 3

Diseñe una clase administradora con soporte para un arreglo de objetos tipo TV. Considere una GUI adecuada para los datos de la clase TV que se indica a continuación.

Diseñe la clase TV con los siguientes atributos: serie (cadena), marca (entero), tamaño en pulgadas(entero), precio en dólares (real). Considere un constructor explícito y sus métodos get/set. Considere un método adicional que devuelve el precio en soles dado el tipo de cambio como parámetro, y un método adicional que retorne el nombre de la marca. Considere las siguientes marcas: Sony, LG, Samsung, Panasonic, otro.

Diseñela clase administradora **ArregloTV**, que tenga como atributo privado un objeto de la clase **ArrayList** particularizado para la clase TV donde se guardarán los objetos.

Desarrolle los siguientes métodos adicionales en la clase administradora:

a) Un método que retorne la cantidad de televisores que pertenecen a una marca específica dada como parámetro.

b) Un método que retorne el precio promedio de los televisores de una marca específica dada como parámetro.

c) Un método que modifique el precio de todos los televisores incrementándolo en un porcentaje dado como parámetro.

En la clase de la GUI considere como atributo un objeto de tipo ArregloTV y programe la acción de los botones de acuerdo a lo siguiente:

- El botón Nuevo permite registrar un nuevo televisor evitando que se repita la serie.
- El botón Busca ubica un televisor según la serie ingresada y muestra sus datos donde corresponda.
- El botón Modifica reemplaza los datos del televisor ubicado con el botón Busca.
- El botón Elimina saca del arreglo el televisor según la serie ingresada.
- El botón Lista muestra una relación de todos los televisores guardados en el arreglo.
- El botón Reporte muestra el resultado de todos los métodos adicinales.
- El botón AumentaPrecio aplica el resultado del método c) y muestra el resultado del botón Lista.
- El botón Borrar limpia todo el contenido del GUI.

Ejemplo 4

Modifique el contenido de la clase **ArregloProductos**, para que considere los siguientes métodos adicionales:

a) Incremente el precio de todos los productos en 15%.

b) Incremente el precio de los productos cuyo precio actual sea inferior a un valor dado como parámetro.

c) Disminuya el precio de los productos cuyo precio actual esté en un rango dado como parámetro.

d) Retorne la cantidad de productos cuyo precio sea inferior al precio promedio.

e) Retorne en un arreglo los productos cuya descripción empiece con una letra dada como parámetro.

Implemente, en la interfaz gráfica de usuario (GUI), lo necesario para llamar a los métodos adicionales desarrollados.

Ejemplo 5

Modifique el contenido de la clase **ArregloPersonas**, para que considere los siguientes métodos adicionales:

a) Incremente el peso de todas las personas en 5%

b) Incremente el peso de las personas cuyo peso actual sea inferior a un valor dado como parámetro.

c) Disminuya el peso de las personas cuyo peso actual esté en un rango dado como parámetro.

d) Retorne la cantidad de personas menores de edad

e) Retorne en un arreglo las personas cuya edad sea superior a un valor dado como parámetro.

Implemente, en la interfaz gráfica de usuario (GUI), lo necesario para llamar a los métodos adicionales desarrollados.

Ejemplo 6

Modifique el contenido de la clase **ArregloTV**, que considere los siguientes métodos adicionales:

a) Incremente el precio de todos los televisores en 8%

b) Incremente el precio de los televisiores de una marca dada como parámetro en un porcentaje también dado como parámetro

c) Disminuya el precio de los televisores cuyo tamaño actual esté en un rango dado como parámetro

d) Retorne la cantidad de televisores de una marca y tamaño dado como parámetros.

e) Retorne en un arreglo los televisores de una marca dada como parámetro.

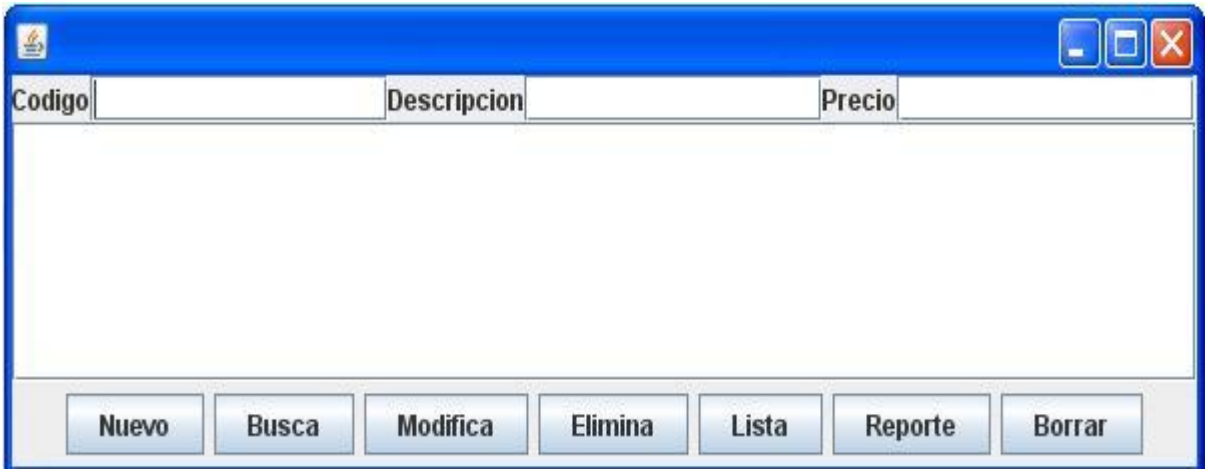
Implemente, en la interfaz gráfica de usuario (GUI), lo necesario para llamar a los métodos adicionales desarrollados.

GUIA DE LABORATORIO 2

Clases administradoras de objetos con ArrayList

Ejercicio 1:

Cree un proyecto nuevo de nombre P02E01. Cree un paquete nuevo de nombre **controlador** y diseñe una clase administradora para un arreglo de objetos de tipo Producto considerando la siguiente GUI:



En el paquete **modelo** diseñe la clase **Producto** con constructor explícito y genere los métodos get-set:

```
public class Producto {
    // atributos privados
    private String codigo,
    descripcion; private double
    precio;

    // constructor
     public Producto(String codigo, String descripcion, double
        precio){ this.codigo=codigo;
        this.descripcion=descripcion;
        this.precio=precio;
    }
    // métodos get-set
}
```

En e paquete **controler** diseñe la clase **ArregloProductos**, que tenga como atributo privado un objeto de la clase **ArrayList** donde se guardarán los objetos y un constructor explícito sin parámetros.

```
public class ArregloProductos {
    // atributos
    // objeto  ArrayList particularizado para objetos de tipo
Producto
    private ArrayList <Producto> a ;

    // constructor explícito
    public ArregloProductos(){
        a = new ArrayList<Producto>();
    }
    // métodos de administración
    public int tamaño(){ return a.size(); }
    public void agrega(Producto p){
      a.add(p);
    }
    public Producto obtiene(int i){ // retorna un producto de la
  posición i
        return a.get(i);
    }
    public void actualiza(Producto p, int i){
      // reemplaza un producto
      a.set(i, p);
    }
    // retorna la posición de un producto según
    su código. public int busca(String codigo){
        for(int i=0; i<tamaño(); i++){
            if(obtiene(i).getCodigo().equals(codigo)
            )
                return i;
        }
        return -1;  // no lo encontró
    }
    public void elimina(int p){// elimina el producto de la
      posición p a.remove(p);
```

```
    }
    public double mayorPrecio(){// retorna el mayor
        precio double m=a.get(0).getPrecio();
        for(Producto p: a){ // for each: por cada producto p en a
            if(p.getPrecio() > m)
                m =p.getPrecio();
        }
        return m;
    }
    public double menorPrecio(){// retorna el menor
        precio double m=a.get(0).getPrecio();
        for(Producto p: a){ // for each: por cada producto p en a
            if(p.getPrecio() < m)
                m =p.getPrecio();
        }
       return m;
    }
    public double precioPromedio(){// retorna el precio promedio
        double suma=0;
        for(Producto p: a){ // for each: por cada producto p en a
            suma += p.getPrecio();
        }
       return suma / tamaño();
    }
} // fin de la clase ArregloProductos
```

Cree un paquete nuevo de nombre **vista** y diseñe la GUI en un Panel de nombre **PanelPrincipal** con un atributo de tipo ArregloProductos y programe la acción de los botones de acuerdo a lo siguiente:

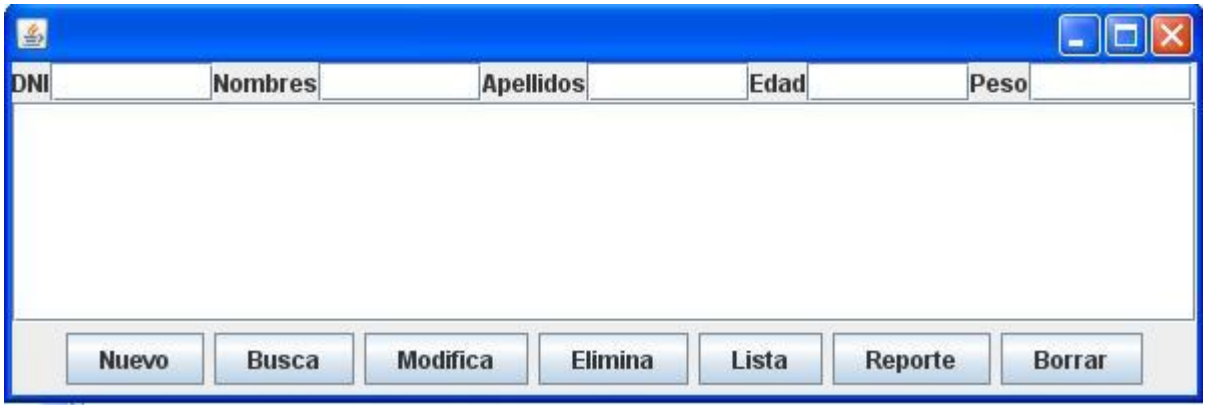
- El botón Nuevo permite registrar un nuevo producto evitando que se repita el código
- El botón Busca ubica un producto según el código ingresado y muestra sus datos donde corresponda.
- El botón Modifica reemplaza los datos del producto ubicado con el botón Busca.
- El botón Elimina saca del arreglo el producto según el código ingresado.
- El botón Lista muestra una relación de todos los productos guardados en el arreglo.

- El botón Reporte muestra los precios mayor, menor y promedio.
- El botón Borrar limpia todo el contenido del GUI.

En el mismo paquete **vista** desarrolle la clase de aplicación de nombre **Principal** en un JFrame. Ejecute su aplicación.

Ejercicio 2:

Cree un proyecto nuevo de nombre P02E02. Cree un paquete nuevo de nombre **controlador** y diseñe una clase administradora para un arreglo de objetos de tipo Persona considerando la siguiente GUI:



En el paquete **modelo** diseñe la clase **Persona** con un constructor explícito:

```
public class Persona {
    // atributos privados
    private String dni, nombres,
    apellidos; private int edad;
    private double peso;

    // constructor explícito
    public Persona(String dni, String nombres, String apellidos,
int edad, double peso){
       this.dni=dni;
        this.nombres=nombres;
        this.apellidos=apellidos;
        this.edad=edad;
        this.peso=peso;
    }
```

```
        // métodos get-set
    }
```

En el paquete **controlador** diseñe la clase administradora **ArregloPersonas**, que tenga como atributo privado un objeto de la clase **ArrayList** particularizado para la clase **Persona**, donde se guardarán los objetos.

Desarrolle los siguientes métodos adicionales en la clase administradora:

a) Un método que retorne el peso mayor de todas las personas.
b) Un método que retorne el peso menor de todas las personas.
c) Un método que retorne el peso promedio de todas las personas.
d) Un método que retorne la cantidad de personas cuyo peso se encuentra en un rango dado como parámetros
e) Un método que retorne la cantidad de personas mayores de edad.
f) Un método que retorne la cantidad de personas menores de edad.

Cree un paquete nuevo de nombre **vista** y diseñe la GUI en un Panel de nombre **PanelPrincipal** con un atributo de tipo ArregloPersonas y programe la acción de los botones de acuerdo a lo siguiente:

- El botón Nuevo permite registrar una nueva persona evitando que se repita el dni
- El botón Busca ubica una persona según el dni ingresado y muestra sus datos donde corresponda.
- El botón Modifica reemplaza los datos de la persona ubicada con el botón Busca.
- El botón Elimina saca del arreglo la persona según el dni ingresado.
- El botón Lista muestra una relación de todas las personas guardadas en el arreglo.
- El botón Reporte muestra el resultado de todos los métodos adicionales.
- El botón Borrar limpia todo el contenido del GUI.

En el paquete **vista** desarrolle la clase de aplicación de nombre **Principal** en un JFrame. Ejecute su aplicación.

Ejercicio 3

Cree un proyecto nuevo de nombre P02E03. Cree un paquete nuevo de nombre **controlador** y diseñe una clase administradora para un arreglo de objetos de tipo TV. Cree un paquete nuevo de nombre **vista** y considere una GUI adecuada para los datos de la clase TV.

En el paquete **modelo** diseñe la clase TV con los siguientes atributos: serie (cadena), marca (entero), tamaño en pulgadas (entero), precio en dólares (real). Considere un constructor explícito y sus métodos get/set. Considere un método adicional que devuelve el precio en soles dado el tipo de cambio como parámetro, y un método adicional que retorne el nombre de la marca. Considere las siguientes marcas: Sony, LG, Samsung, Panasonic, otro.

En el paquete **controlador** diseñe la clase administradora **ArregloTV**, que tenga como atributo privado un objeto de la clase **ArrayList** particularizado para la clase TV donde se guardarán los objetos.

Desarrolle los siguientes métodos adicionales en la clase administradora:

a) Un método que retorne la cantidad de televisores que pertenecen a una marca específica dada como parámetro.

b) Un método que retorne el precio promedio de los televisores de una marca específica dada como parámetro.

c) Un método que modifique el precio de todos los televisores incrementándolo en un porcentaje dado como parámetro.

En el paquete **vista** Diseñe la GUI en un Panel de nombre **PanelPrincipal** con un atributo de tipo ArregloTV y programe la acción de los botones de acuerdo a lo siguiente:

- El botón Nuevo permite registrar un nuevo televisor evitando que se repita la serie.
- El botón Busca ubica un televisor según la serie ingresada y muestra sus datos donde corresponda.

- El botón Modifica reemplaza los datos del televisor ubicado con el botón Busca.
- El botón Elimina saca del arreglo el televisor según la serie ingresada.
- El botón Lista muestra una relación de todos los televisores guardados en el arreglo.
- El botón Reporte muestra el resultado de todos los métodos adicinales.
- El botón AumentaPrecio aplica el resultado del método c) y muestra el resultado del botón Lista.
- El botón Borrar limpia todo el contenido del GUI.

En el paquete vista desarrolle la clase de aplicación de nombre **Principal** en un JFrame. Ejecute su aplicación.

Ejercicio 4

Modifique el contenido de la clase **ArregloProductos**, para que considere los siguientes métodos adicionales:

a) Incremente el precio de todos los productos en 15%.

b) Incremente el precio de los productos cuyo precio actual sea inferior a un valor dado como parámetro.

c) Disminuya el precio de los productos cuyo precio actual esté en un rango dado como parámetro.

d) Retorne la cantidad de productos cuyo precio sea inferior al precio promedio.

e) Retorne en un arreglo los productos cuya descripción empiece con una letra dada como parámetro.

Implemente, en la interfaz gráfica de usuario (GUI), lo necesario para llamar a los métodos adicionales desarrollados.

Ejercicio 5

Modifique el contenido de la clase **ArregloPersonas**, para que considere los siguientes métodos adicionales:

a) Incremente el peso de todas las personas en 5%

b) Incremente el peso de las personas cuyo peso actual sea inferior a un valor dado como parámetro.

c) Disminuya el peso de las personas cuyo peso actual esté en un rango dado como parámetro.

d) Retorne la cantidad de personas menores de edad

e) Retorne en un arreglo las personas cuya edad sea superior a un valor dado como parámetro.

Implemente, en la interfaz gráfica de usuario (GUI), lo necesario para llamar a los métodos adicionales desarrollados.

Ejercicio 6

Modifique el contenido de la clase **ArregloTV**, que considere los siguientes métodos adicionales:

a) Incremente el precio de todos los televisores en 8%

b) Incremente el precio de los televisiores de una marca dada como parámetro en un porcentaje también dado como parámetro

c) Disminuya el precio de los televisores cuyo tamaño actual esté en un rango dado como parámetro

d) Retorne la cantidad de televisores de una marca y tamaño dado como parámetros.

e) Retorne en un arreglo los televisores de una marca dada como parámetro.

Implemente, en la interfaz gráfica de usuario (GUI), lo necesario para llamar a los métodos adicionales desarrollados.