EJERCICIO GUIADO. JAVA: VECTORES

Vectores en Java

El manejo de vectores en Java es similar al manejo de vectores en C. Solo hay que tener en cuenta que un vector en Java se tiene primero que *declarar*, y luego se tiene que *construir*, ya que los vectores son, al igual que todo en Java, objetos.

Declaración de un vector

Para declarar un vector se seguirá la siguiente sintaxis:

```
tipodatos  nombrevector[];
```

- `tipodatos` es el tipo de datos de los elementos del vector (int, double, String, etc.)
- `nombrevector` es el nombre que tu quieras darle a tu vector.

Por ejemplo, supongamos que queremos crear un vector de enteros llamado v. La declaración será la siguiente:

```
int v[];
```

Como puedes ver, en la declaración de un vector no se indica la dimensión, de éste, es decir, el número de elementos que tiene. Esto se hace en la *construcción* del vector.

Construcción de un vector

Para construir un vector que ya haya sido declarado se sigue la siguiente sintaxis:

```
nombrevector = new tipodatos[dim];
```

- `nombrevector` es el nombre del vector a construir.
- `tipodatos` es el tipo de datos del vector.
- `dim` es la dimensión del vector (el número de elementos)

Por ejemplo, si queremos que el vector v declarado antes tenga 10 elementos, tendremos que hacer lo siguiente:

```
v = int[10];
```

En el momento de la construcción del vector, podemos usar una variable entera para asignar el número de elementos que se quiera.

Por ejemplo, en el siguiente caso el número de elementos del vector v viene dado por la variable *num*:

```
v = int[num];
```

Acceso a los elementos de un vector

Una vez declarado el vector y construido, este se puede usar de la misma manera que en C. Se puede acceder a un elemento del vector escribiendo el nombre del vector y entre corchetes el índice del elemento que quieres usar. Recuerda que los índices comienzan a numerarse en 0.

Ejemplo 1:

```
etiResultado.setText(“El resultado es: “+v[3]);
```

Aquí se coloca en una etiqueta el valor contenido en el elemento de la posición cuarta del vector v.

Ejemplo 2:

```
for (i=0;i<10;i++) {
   v[i]=0;
}
```

Este código recorre un vector de 10 elementos y almacena en dichos 10 elementos un 0.

Longitud de un vector

Una forma rápida de saber la longitud que tiene un vector es usar lo siguiente:

```
nombrevector.length
```

Por ejemplo, si el vector se llama v, su longitud (el número de elementos que tiene) sería:

```
v.length
```

El siguiente código rellena el vector v con ceros, da igual el número de elementos que tenga:

```
for (i=0;i<v.length;i++) {
      v[i]=0;
}
```

Creación de un vector e inicialización con datos al mismo tiempo

Es posible crear un vector e introducir datos directamente en él al mismo tiempo. La forma general de hacerlo sería la siguiente:

```
tipodatos nombrevector[] = {elemento1, elemento2, elemento3, …, elemento n};
```

Por ejemplo:

```
int v[] = {5, 2, 7, 6};
```

Este código crea un vector con cuatro números enteros: 5, 2, 7, 6.

Ejemplo 2:

```
String dias[] = {“Lunes”, “Martes”, “Miércoles”, “Jueves”, “Viernes”, “Sábado”, “Domingo”};
```

Este código crea un vector que contiene 7 cadenas, correspondientes a los días de la semana.

VECTORES DE COMPONENTES

En Java, se pueden crear vectores de etiquetas, botones, cuadros de textos, etc.

Esto es tremendamente útil cuando se tienen que definir muchos componentes con una función parecida, y que tengan que ser tratados en conjunto.

Por otro lado, nos ahorra mucho tiempo ya que no se tienen que diseñar estos elementos en la misma ventana de diseño.

Los vectores de componentes se usan igual que se ha indicado antes. Observa el siguiente ejemplo:

```
JLabel veti[];  //aquí se crea un vector de etiquetas llamado veti

veti=new JLabel[3]; //aquí se construye el vector, asignando 3 etiquetas

//ahora trabajamos con las etiquetas del vector
//usando el típico for

for (i=0;i<veti.length;i++) {
      veti[i]= new JLabel(); //se construye cada etiqueta
      veti[i].setBounds(10,10+i*30,100,20); //se asigna posición y tamaño
      veti[i].setText(“Etiqueta “+i); //se asigna un texto
      this.getContentPane().add(veti[i]); //se coloca en la ventana
}
```

Realiza el siguiente ejercicio guiado para entender el ejemplo:

Ejercicio guiado

1. Crea un nuevo proyecto en java.

2. Empecemos definiendo desde código las características de la ventana. Crearemos el método *CreacionVentana*, y lo llamaremos desde el constructor del proyecto:

```
public ventanaprincipal() {
    initComponents();
    CreacionVentana();
}

public void CreacionVentana() {

    //Caracteristicas de la ventana
    this.setTitle("Ejemplo de vector de componentes");
    this.setSize(300,600);
}
```

3. Vamos a situar en la ventana 10 cuadros de verificación (JCheckBox). Para ello, usaremos un vector de 10 JCheckBox. Este vector se declarará en la zona de variables globales (será necesario añadir el típico *import*):

```
public class ventanaprincipal extends javax.swing.JFrame {

    JCheckBox vcuadros[];

    /** Creates new form ventanaprincipal */
    public ventanaprincipal() {
        initComponents();
        CreacionVentana();
    }

    public void CreacionVentana() {

        //Caracteristicas de la ventana
        this.setTitle("Ejemplo de vector de componentes");
        this.setSize(300,600);
    }
```

4. La construcción de los vectores de componentes, se realiza en el mismo constructor (en nuestro caso en el método *CreaciónVentana*, que es llamado desde el constructor). Construiremos el vector de forma que contenga 10 JCheckBox:

```
public class ventanaprincipal extends javax.swing.JFrame {

    JCheckBox vcuadros[];

    /** Creates new form ventanaprincipal */
    public ventanaprincipal() {
        initComponents();
        CreacionVentana();
    }

    public void CreacionVentana() {

        //Caracteristicas de la ventana
        this.setTitle("Ejemplo de vector de componentes");
        this.setSize(300,600);

        //construccion del vector (10 elementos)
        vcuadros = new JCheckBox[10];
    }
```

5. Se acaba de construir un vector con 10 cuadros de verificación. Ahora, es necesario construir cada uno de los cuadros del vector y asignarle ciertas propiedades. Esto se hace con un for que recorre los elementos del vector. (Añade el siguiente código dentro de *CreacionVentana*):

```
//construcción de cada cuadro de verificación del vector
//y asignación de propiedades a dichos cuadros
int i;
for (i=0;i<vcuadros.length;i++) {
    vcuadros[i]=new JCheckBox(); //se construye cada cuadro
    vcuadros[i].setText("Opción "+i);  //se le da un texto
    vcuadros[i].setBounds(10, 10+30*i, 100,20); //posición y tamaño
    this.getContentPane().add(vcuadros[i]);
}
```

6. Analiza este código que se acaba de añadir:

    a. Observa como cada elemento del vector debe ser construido:

    ```
    vcuadros[i] = new JCheckBox();
    ```

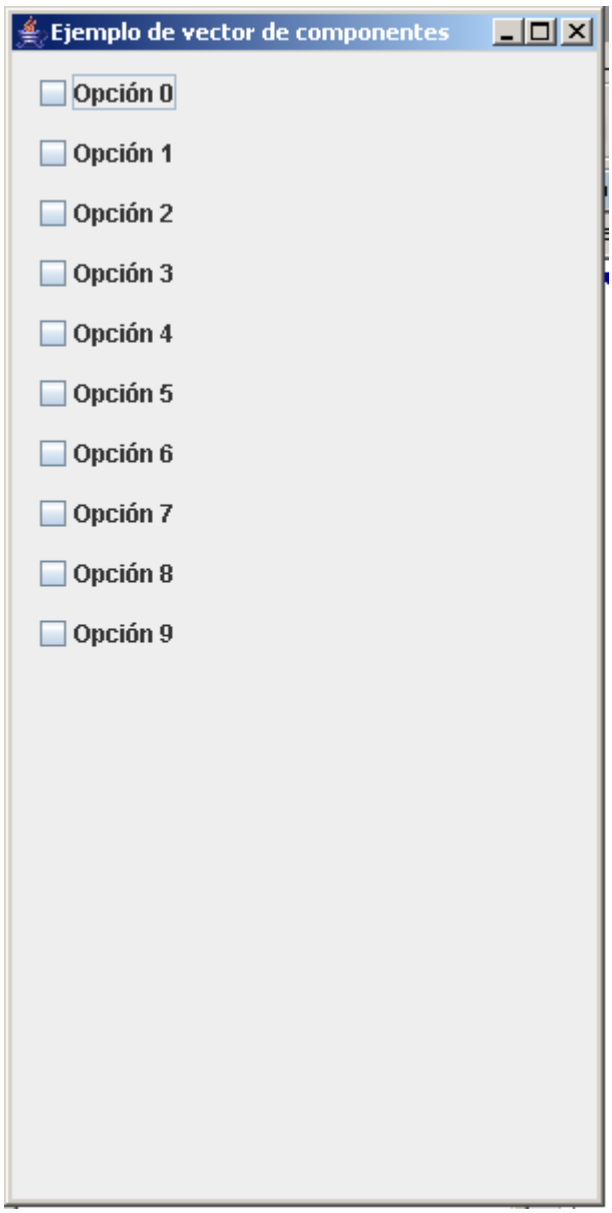
    b. El texto que tendrá cada elemento será: “Opción 0”, “Opción 1”, etc…

    ```
    vcuadros[i].setText(“Opción “+i);
    ```

    c. Los cuadros de verificación se colocan uno debajo de otro. Estudia la línea siguiente y observa como varía la posición vertical de cada cuadro:

    ```
    vcuadros[i].setBounds(10, 10+30*i,  100, 20);
    ```

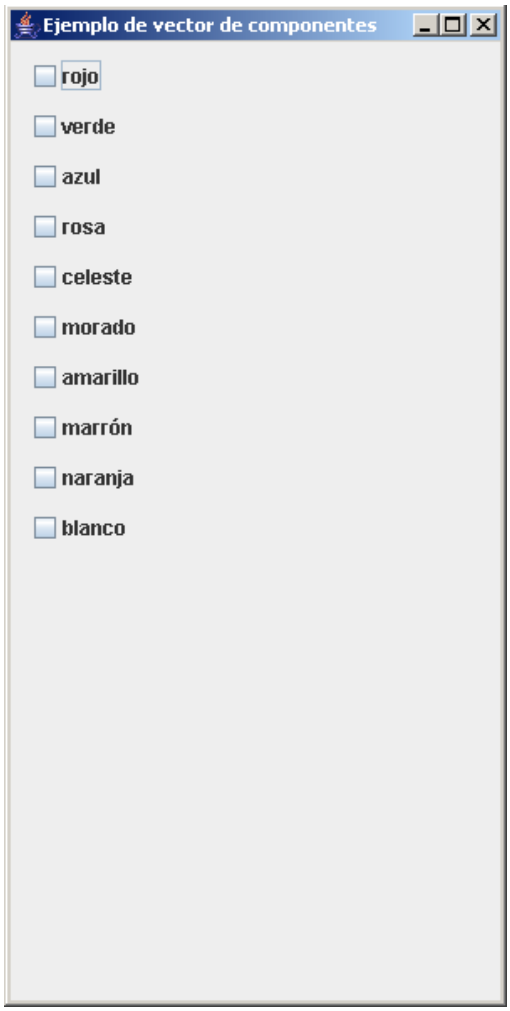
7. Ejecuta el programa y observa el resultado:



8. Mejoremos la presentación de los cuadros cambiando los rótulos de cada uno. Para ello, será necesario usar otro vector auxiliar que contenga los textos de cada uno de los cuadros. Modifica el código anterior de forma que quede así:

```
//construcción de cada cuadro de verificación del vector
//y asignación de propiedades a dichos cuadros
String vtextos[]={"rojo","verde","azul","rosa","celeste",
                  "morado","amarillo","marrón","naranja","blanco"};
int i;
for (i=0;i<vcuadros.length;i++) {
    vcuadros[i]=new JCheckBox(); //se construye cada cuadro
    vcuadros[i].setText(vtextos[i]); //se le da un texto del vector
                                     //de textos
    vcuadros[i].setBounds(10, 10+30*i, 100,20); //posición y tamaño
    this.getContentPane().add(vcuadros[i]);
}
```

9. En este código puedes observar como se usa un vector de String que se crea conteniendo 10 colores. Luego, ese vector se usa para asignar cada color al texto de cada cuadro. Si ahora ejecutas el programa, verás que cada cuadro tiene su texto correspondiente a un color.



10. Ahora añade un botón con su evento *actionPerformed*. Añade en la zona de variables globales lo siguiente:

```
JButton btnAceptar;
```

Y luego, dentro de *CreacionVentana*, añade el siguiente código:

```
btnAceptar = new JButton();
btnAceptar.setText("Aceptar");
btnAceptar.setBounds(10,360,100,20);
this.getContentPane().add(btnAceptar);

btnAceptar.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent evt) {
       btnAceptarActionPerformed(evt);
   }
});
```

11. Ahora programaremos la respuesta al evento *actionPerformed* de forma que el programa diga cuantos cuadros hay seleccionados. Para ello, se tendrá que programar el procedimiento *btnAceptarActionPerformed*, cuya llamada se encuentra en el código anterior (la línea que da error):

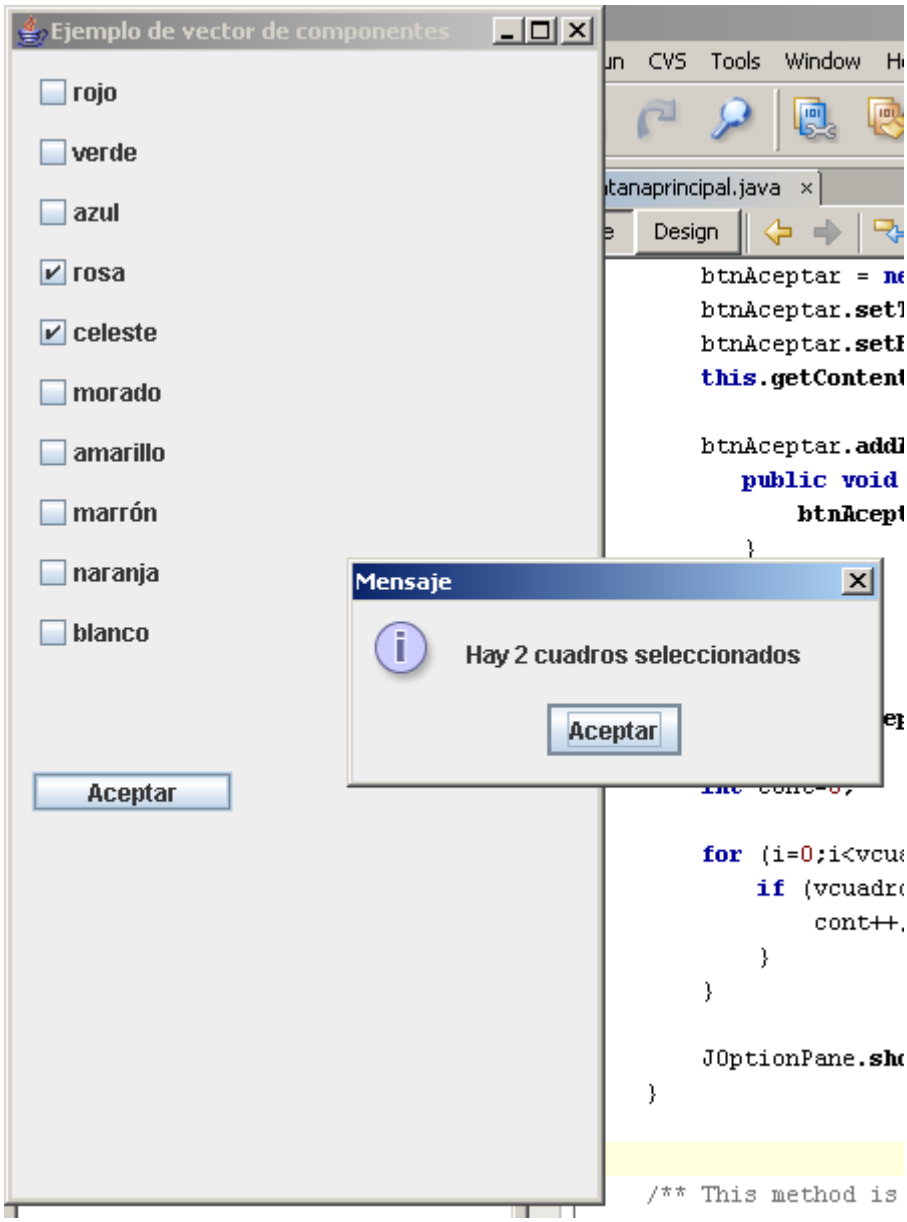
Así pues, fuera de *CreacionVentana*, programa el siguiente procedimiento:

```
public void btnAceptarActionPerformed(ActionEvent evt) {
    int i;
    int cont=0;

    for (i=0;i<vcuadros.length;i++) {
        if (vcuadros[i].isSelected()) {
            cont++;
        }
    }

    JOptionPane.showMessageDialog(null,"Hay "+cont+" cuadros seleccionados");
}
```

12. En este código se puede observar como se usa un for para recorrer fácilmente el vector de cuadros y averiguar cuales de ellos está activado. Aumentamos un contador y lo demás es sencillo.

Si este programa se hubiera hecho desde diseño, el código para contar el número de cuadros activados sería mucho más engorroso. Piénsalo.

13. Ejecuta el programa. Selecciona varios cuadros y pulsa el botón Aceptar. Observa el resultado:



CONCLUSIÓN

En Java, los vectores debe declararse y luego construirse. Es en la construcción del vector cuando a este se le asigna un número de elementos.

Los vectores en Java pueden ser usados de la misma forma que en C.

En Java se pueden crear vectores de componentes: etiquetas, botones, etc, facilitando luego el trabajo con todos estos elementos en conjunto y facilitando las labores de diseño.