Ejercicio 22

```
private void aplicarOscuro()
{
    System.out.println("Oscuro.");
}
ItsIvanPsk *
private void aplicarClaro()
{
    System.out.println("Claro.");
}
ItsIvanPsk *
private void aplicarUmbral()
{
    System.out.println("Umbral.");
}
```

Ejercicio 23

El método aplicar oscuro llama al método .oscuro y al método ventana.repaint
El método .oscuro recorre todos lo pixeles de la imagen abierta y le aplica el filtro "Oscuro" a todos los pixeles por igual. Luego de hacer esto el codigo llama a ventana.repaint() para que se actualice la imagen en la interfaz.

Ejercicio 24

```
JLabel etiquetaEstado;
2 usages
JLabel etiquetaNombreDeArchivo;
```

```
private void mostrarEstado(String str)
{
    etiquetaEstado.setText(str);
}
```

Ejercicio 25

Pues que se actualiza la etiqueta usando el método mostrarEstado en la parte inferior de la ventana.

Ejercicio 26

```
private void oscuro(){
    int alto = getHeight();
    int ancho = getWidth();
    for (int i = 0; i < alto; i++){
        for (int j = 0; j < ancho; j++){
            setPixel(x,y,getPixel(x,y).darker());
        }
    }
}
```

Este método lo que hace es, primero setea dos ints para poder saber que tan grande es la imagen que pretendemos abrir.
Luego recorremos todos los píxeles de la imagen para poder obtener información del mismo, luego seteamos el pixel pasándole por parámetros, la posición del píxel usando las dos variables contador de los for's (i y j) y luego le aplicamos el método darker de la clase Color para modificar el color de la imagen.

El método darker es de la clase Color.

Ejercicio 27

```
new *
private void claro(){
    int alto = getHeight();
    int ancho = getWidth();
    for (int i = 0; i < alto; i++){
        for (int j = 0; j < ancho; j++){
            setPixel(x,y,getPixel(x,y).brighter());
        }
    }
}
```

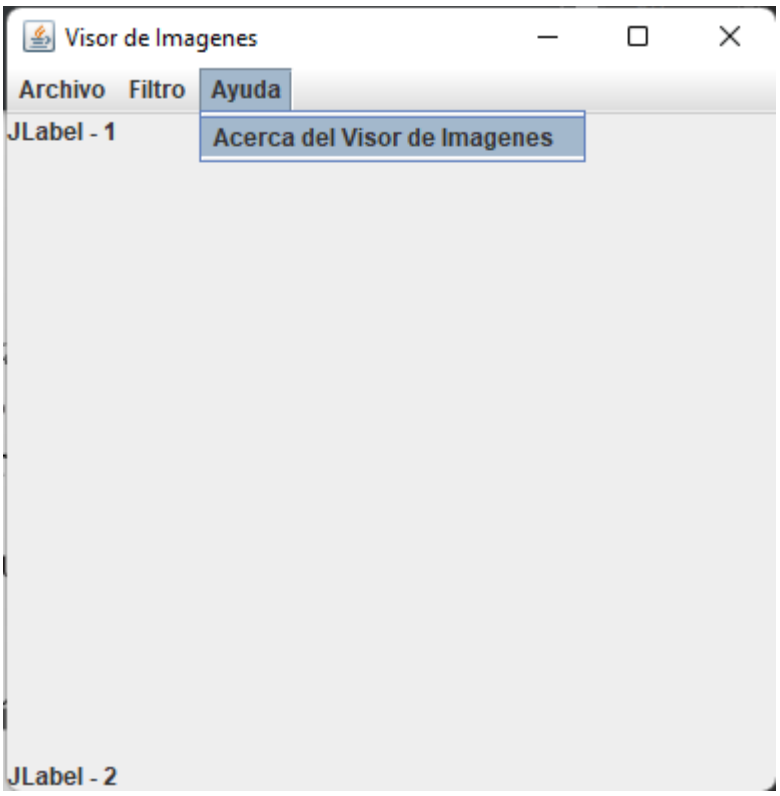
Usando el método brighter en vez del método darker, igualmente de la clase Color.

Ejercicio 28

```java
public void umbral()
{
    int alto = getHeight();
    int ancho = getWidth();
    for(int y = 0; y < alto; y++) {
        for(int x = 0; x < ancho; x++) {
            Color pixel = getPixel(x, y);
            int brillo = (pixel.getRed() + pixel.getBlue() + pixel.getGreen()) / 3;
            if(brillo <= 85) {
                setPixel(x, y, Color.BLACK);
            }
            else if(brillo <= 170) {
                setPixel(x, y, Color.GRAY);
            }
            else {
                setPixel(x, y, Color.WHITE);
            }
        }
    }
}
```

Ejercicio 29

```java
JMenu menuAyuda = new JMenu( s: "Ayuda");
barraDeMenu.add(menuAyuda);
JMenuItem elementoAcercaDe = new JMenuItem( text: "Acerca del Visor de Imagenes");
```



Ejercicio 30

```
elementoAcercaDe.addActionListener(new ActionListener() {
    ItsIvanPsk
    public void actionPerformed(ActionEvent e) { acercaDe(); }
});
menuAyuda.add(elementoAcercaDe);
```

```
private void acercaDe() {
    System.out.println("Ayuda");
}
```

Ejercicio 31

showConfirmDialog: *Muestra una ventana en la que se tiene un boton, se usa para confirmar, o aceptar algo.*

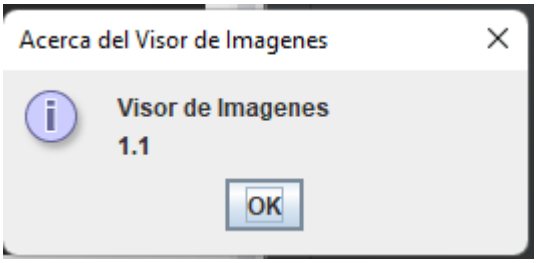
showInputDialog: *Muestra una ventana en la que se tiene un inputText, se usa para poder rellenar con información del usuario por tal de poder hacer que el usuario pueda ingresar datos de una forma más cómoda.*

showMessageDialog: *Muestra una ventana en la que se tiene un mensaje, se usa solamente para darle una información al usuario y luego se cierra.*

showOptionDialog: *Muestra una ventana en la que se tiene un listado de opciones que el usuario deberá de marcar.*

Ejercicio 32

```
private void mostrarAcerdaDe(){
    JOptionPane.showMessageDialog(ventana,
            message: "Visor de Imagenes\n" + VERSION,
            title: "Acerca del Visor de Imagenes",
            JOptionPane.INFORMATION_MESSAGE);
}
```



Ejercicio 33

Se puede bloquear que el usuario pueda escribir en el JTextField con la siguiente linea de comando: **.setEditable(false);.**

Podemos saber si el usuario está cambiando el contenido del JTextField desde un listener llamado addKeyListener desde su método keyPressed. Imagen ejemplo código:

```
KeyListener keyListener = new KeyListener() {
  public void keyPressed(KeyEvent keyEvent) {
    printIt("Pressed", keyEvent);
  }

  public void keyReleased(KeyEvent keyEvent) {
    printIt("Released", keyEvent);
  }

  public void keyTyped(KeyEvent keyEvent) {
    printIt("Typed", keyEvent);
  }

  private void printIt(String title, KeyEvent keyEvent) {
    int keyCode = keyEvent.getKeyCode();
    String keyText = KeyEvent.getKeyText(keyCode);
    System.out.println(title + " : " + keyText + " / " + keyEvent.getKeyChar());
  }
};
```