```java
package E705;

import java.awt.EventQueue;
import java.awt.Graphics;
import java.awt.Image;
import java.awt.color.ColorSpace;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.image.BufferedImage;
import java.awt.image.ColorConvertOp;
import java.awt.image.WritableRaster;
import java.io.File;
import java.io.IOException;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JButton;
import javax.swing.JFileChooser;

import java.awt.BorderLayout;

import javax.imageio.ImageIO;
import javax.swing.ImageIcon;
import javax.swing.JScrollPane;
import javax.swing.border.LineBorder;
import javax.swing.filechooser.FileNameExtensionFilter;

import java.awt.Color;
import javax.swing.JPanel;

public class E705 {

    private JFrame ecualizador;                 //JFrame
    private JPanel pCopia;                      //Jpanel donde metemeos el histograma de la copia
    private File fichero;                       //para guardar el fichero que leemos
    private JScrollPane SPfotoOriginal;         //ScrollPane de la foto original
    private JScrollPane SPfotoCopia;            //ScrollPane de la foto que es copia
    private BufferedImage bi;                   //imagen con la que vamos a trabajar
    private ImageIcon ii;                       //paso intermedio de file a bufferedImage
    private double[]histo = new double[256];    //histograma de los pixeles
    private double[]histoAcu = new double[256]; //histrograma acumulado
    private JPanel pOriginal;                   //Jpanel donde metemos el histograma de la foto original

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    E705 window = new E705();
```

```
52                      window.ecualizador.setVisible(true);
53                  } catch (Exception e) {
54                      e.printStackTrace();
55                  }
56              }
57          });
58      }
59
60      /**
61       * Create the application.
62       */
63      public E705() {
64          initialize();
65      }
66
67      /**
68       * Initialize the contents of the frame.
69       */
70      private void initialize() {
71          ecualizador = new JFrame();
72          ecualizador.setTitle("Jorge Victoria Andreu");
73          ecualizador.setBounds(50, 50, 800, 700);
74          ecualizador.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
75          ecualizador.getContentPane().setLayout(null);
76
77          JButton btAbrir = new JButton("Abrir");
78          btAbrir.setIcon(new ImageIcon("C:\\Users\\jvand\\OneDrive\\Estudios\\Workspace\\Eclipse\\Tema
            07\\src\\E705\\iconos\\carpeta.png"));
79          btAbrir.setBounds(10, 11, 120, 23);
80          ecualizador.getContentPane().add(btAbrir);
81
82          JButton btEcualizar = new JButton("Ecualizar");
83          btEcualizar.setIcon(new ImageIcon("C:\\Users\\jvand\\OneDrive\\Estudios\\Workspace\\Eclipse\\Tema
            07\\src\\E705\\iconos\\ecualizador.png"));
84          btEcualizar.setEnabled(false);
85          btEcualizar.setBounds(140, 11, 120, 23);
86          ecualizador.getContentPane().add(btEcualizar);
87
88          JButton btGuardar = new JButton("Guardar");
89          btGuardar.setEnabled(false);
90          btGuardar.setIcon(new ImageIcon("C:\\Users\\jvand\\OneDrive\\Estudios\\Workspace\\Eclipse\\Tema
            07\\src\\E705\\iconos\\disquette.png"));
91          btGuardar.setBounds(270, 11, 120, 23);
92          ecualizador.getContentPane().add(btGuardar);
93
94          SPfotoCopia = new JScrollPane();
95          SPfotoCopia.setBounds(404, 45, 370, 255);
96          ecualizador.getContentPane().add(SPfotoCopia);
97
98          SPfotoOriginal = new JScrollPane();
99          SPfotoOriginal.setBounds(10, 45, 370, 255);
```

```
100             ecualizador.getContentPane().add(SPfotoOriginal);
101
102             pOriginal = new JPanel() {
103
104                 public void paint(Graphics g) {
105                     super.paint(g);
106                     g.setColor(Color.black);
107
108                     //ejes horizontal y vertical
109                     g.drawLine(30, 270, 286, 270);
110                     g.drawLine(30, 270, 30, 20);
111
112                     g.drawString("0", 28, 290 );
113
114                     int ancho = 28;
115                     //dibujamos marcas y numeros
116                     for(int i = 1; i <= 5; i++) {
117                         ancho = ancho + 50;
118                         g.drawString(String.valueOf(i*50), ancho-5, 290);
119                         g.drawLine(ancho, 268, ancho, 272);
120                     }
121                 }
122             };
123
124             pOriginal.setBorder(new LineBorder(new Color(0, 0, 0)));
125             pOriginal.setBackground(Color.WHITE);
126             pOriginal.setBounds(10, 311, 370, 300);
127             ecualizador.getContentPane().add(pOriginal);
128
129             pCopia = new JPanel() {
130                 public void paint(Graphics g) {
131                     super.paint(g);
132                     g.setColor(Color.black);
133
134                     //ejes horizontal y vertical
135                     g.drawLine(30, 270, 286, 270);
136                     g.drawLine(30, 270, 30, 20);
137
138                     g.drawString("0", 28, 290 );
139
140                     int ancho = 28;
141                     //dibujamos marcas y numeros
142                     for(int i = 1; i <= 5; i++) {
143                         ancho = ancho + 50;
144                         g.drawString(String.valueOf(i*50), ancho-5, 290);
145                         g.drawLine(ancho, 268, ancho, 272);
146                     }
147                 }
148             };
149             pCopia.setBorder(new LineBorder(new Color(0, 0, 0)));
150             pCopia.setBackground(Color.WHITE);
```

```
151             pCopia.setBounds(404, 311, 370, 300);
152             ecualizador.getContentPane().add(pCopia);
153 
154             //listener para el jfilechooser
155             btAbrir.addActionListener(new ActionListener() {
156                 public void actionPerformed(ActionEvent e) {
157                     //buscamos el fichero
158                     fichero = leerFichero();
159 
160                     //cargamos la imagen
161                     cargaImagen();
162 
163                     //vamos a crear el histograma
164                     calculoHistograma();
165 
166                     //vamos a pintar el histograma
167                     pintaGrafica(pOriginal);
168 
169                     //ponemos el boton de ecualizar activo
170                     btEcualizar.setEnabled(true);
171             }
172             });
173 
174             //listener para el ecualizador
175                     btEcualizar.addActionListener(new ActionListener() {
176                         public void actionPerformed(ActionEvent e) {
177                             //ecualizamos la imagen
178                             bi = ecualizar(bi);
179 
180                             //cargamos la imagen
181                             cargaImagenCopia();
182 
183                             //vamos a crear el histograma
184                             calculoHistograma();
185 
186                             //vamos a pintar el histograma
187                             pintaGrafica(pCopia);
188 
189                             //ponemos el boton de guardar activo
190                             btGuardar.setEnabled(true);
191                     }
192                     });
193 
194                     //listener para el guardado
195                     btGuardar.addActionListener(new ActionListener() {
196                         public void actionPerformed(ActionEvent e) {
197                             guardarFichero();
198                     }
199                     });
200         }
201 
```

```
202     //metodo para leer fichero con jfileChooer
203     public static File leerFichero() {
204 
205         JFileChooser fd = new JFileChooser();
206         fd.setAcceptAllFileFilterUsed(false);
207         FileNameExtensionFilter filter = new FileNameExtensionFilter("archivos imagen", "jpg", "png", "gif");
208         fd.addChoosableFileFilter(filter);
209         fd.setDialogTitle("Selecciona el fichero a leer");
210         fd.setSelectedFile(null);
211         int opcion = fd.showOpenDialog(null);
212 
213         if (opcion != JFileChooser.APPROVE_OPTION) return null;
214 
215         File f = fd.getSelectedFile();
216 
217         return f;
218 
219     }
220 
221     //metodo para cargar la imagen en pantalla
222     public void cargaImagen() {
223 
224         try {
225             bi = ImageIO.read(fichero);
226         } catch (IOException e1) {
227             // TODO Bloque catch generado automáticamente
228             e1.printStackTrace();
229         }
230 
231         //llamamos al metodo para pasar a escala de grises
232         bi = escalarGrises(bi);
233 
234         //escalamos la imagen
235         ii = new ImageIcon(bi.getScaledInstance(240, 291, Image.SCALE_SMOOTH));
236 
237         //construimos un jlabel para poder incrustar la imagen en el jpanel
238         JLabel etiqueta = new JLabel(ii);
239 
240         //cargamos la foto en el jscrollpane
241         SPfotoOriginal.setViewportView(etiqueta);
242     }
243 
244     //metodo para cargar la imagen ecualizada en pantalla
245     public void cargaImagenCopia() {
246 
247         //escalamos la imagen
248         ii = new ImageIcon(bi.getScaledInstance(240, 291, Image.SCALE_SMOOTH));
249 
250         //construimos un jlabel para poder incrustar la imagen en el jpanel
251         JLabel etiqueta = new JLabel(ii);
252 
```

```
253         //cargamos la foto en el jscrollpane
254         SPfotoCopia.setViewportView(etiqueta);
255     }
256 
257     //metodo para escalar la imagen a grises, si viene en color
258     public BufferedImage escalarGrises(BufferedImage imagen) {
259         int mediaPixeles;
260         int colorSRGB;
261         Color nuevoColor;
262 
263          //Recorremos la imagen píxel a píxel
264         for( int i = 0; i < imagen.getWidth(); i++ ){
265             for( int j = 0; j < imagen.getHeight(); j++ ){
266                 //Almacenamos el color del píxel
267                 nuevoColor=new Color(imagen.getRGB(i, j));
268                 //Calculamos la media de los tres canales (rojo, verde, azul)
269                 mediaPixeles=(int)((nuevoColor.getRed()+nuevoColor.getGreen()+nuevoColor.getBlue())/3);
270                 //Cambiamos a formato sRGB
271                 colorSRGB=(mediaPixeles << 16) | (mediaPixeles << 8) | mediaPixeles;
272                 //Asignamos el nuevo valor al BufferedImage
273                 imagen.setRGB(i, j,colorSRGB);
274             }
275         }
276         //Retornamos la imagen
277         return imagen;
278     }
279 
280     //metodo para calcular los histogramas
281     public void calculoHistograma() {
282         int nivel;
283         //ponemos el array de colores a 0
284         for(int i = 0; i < 256; i++) {
285             histo[i] = 0;
286         }
287 
288         //Recorremos la imagen píxel a píxel y rellenamos el array histo
289         for( int i = 0; i < bi.getWidth(); i++ ){
290             for( int j = 0; j < bi.getHeight(); j++ ){
291                 Color color =new Color( bi.getRGB(i, j));
292                 nivel = (color.getRed() +color.getGreen() +color.getBlue() )/3;
293                 histo[nivel]++;
294             }
295         }
296 
297         //ahora rellenamos el array acumulado
298         //primero rellenamos el array de 0
299         for(int i = 0; i < 256; i++) {
300             histoAcu[i] = 0;
301         }
302 
303         //ponemos la posicion 0
```

```
304             histoAcu[0] = histo[0];
305 
306             //y ya rellenamos el resto
307             for(int i = 1; i < 256; i++) {
308                 histoAcu[i] = histoAcu[i-1] + histo[i];
309             }
310 
311         }
312 
313         //metodo para pintar las graficas en pantalla
314         public void pintaGrafica(JPanel original ) {
315 
316             Graphics gr = original.getGraphics();
317 
318             gr.setColor(Color.red);
319 
320             //pintamos barra con el resultado de las posiciones del arry histo
321             for(int i = 0; i < 256; i++) {
322                 gr.drawLine(30+i, 270-(int)histo[i]/20, 30+i, 270);
323             }
324 
325             gr.setColor(Color.black);
326 
327             //pintamos linea histograma historico acumulado
328             for(int i = 0; i < 256; i++) {
329                 gr.drawString("-", 30+i, 270-(int)histoAcu[i]/600);
330             }
331 
332         }
333 
334         //metodo para ecualizar la imagen
335         public BufferedImage ecualizar(BufferedImage imagen) {
336             int mediaPixeles;
337             int colorSRGB;
338             Color nuevoColor;
339 
340             for (int i = 0; i < imagen.getWidth(); i++) {
341                 for (int j = 0; j < imagen.getHeight(); j++) {
342                     //Almacenamos el color del píxel
343                     nuevoColor=new Color(imagen.getRGB(i, j));
344                     //Calculamos la media de los tres canales (rojo, verde, azul)
345                     mediaPixeles=(int)((nuevoColor.getRed()+nuevoColor.getGreen()+nuevoColor.getBlue())/3);
346 
347                     imagen.setRGB(i,j,(int) (255*histoAcu[mediaPixeles]/histoAcu[255]));
348                 }
349             }
350             return imagen;
351         }
352 
353 
354         //metodo para guardar el fichero
```

```
355       public void guardarFichero() {
356
357           JFileChooser fd = new JFileChooser();
358           fd.setFileSelectionMode(JFileChooser.FILES_ONLY);
359           fd.setAcceptAllFileFilterUsed(false);
360           fd.setDialogTitle("Guardar imagen");
361           fd.setApproveButtonText("Aceptar");
362           int opcion = fd.showSaveDialog(null);
363           FileNameExtensionFilter filter = new FileNameExtensionFilter("archivos imagen", "jpg");
364           if(opcion==JFileChooser.APPROVE_OPTION){
365               File f = fd.getSelectedFile();
366               String test = f.getAbsolutePath();
367               try {
368                   ImageIO.write(bi,"jpg",new File(test));
369               } catch (IOException e1) {
370                   // TODO Auto-generated catch block
371                   e1.printStackTrace();
372               }
373           }
374       }
375   }
376
```