```java
package efectos;

import imagenes.ColorRGB;

/**
 * Esta clase permite dibujar un marco a la imagen, con el color y grosor indicados.
 *
 *
 */
public class EfectoMarco extends Efecto{

    private ColorRGB colorMarco;
    private int anchoMarco;


    /**
     * Construye un objeto que permitirá dibujar un marco a la imagen.
     *
     * @param nombre Nombre del efecto.
     * @param imagen Imagen a procesar.
     * @param ancho  Anchura del marco en píxeles. Como máximo debe ser la mitad del menor valor entre la altura y la anchura de la imagen. Si se supera este valor, el ancho del marco será colocado a 0.
     * @param color  Color del marco.
     */
    public EfectoMarco(String nombre, ImagenRGB imagen, int ancho , ColorRGB color) {

        super(nombre, imagen);
        colorMarco =  color;

        if(ancho < 0 ) {
            this.anchoMarco = 0;
            throw new IllegalArgumentException ("El parámetro ancho es menor que 0 ");

        }if(ancho > imagen.getAlto()/2 && ancho > imagen.getAncho()/2 ) {

            this.anchoMarco = 0;
            throw new IllegalArgumentException ("El parámetro ancho es  mayor que la mitad del menor valor entre el ancho y el alto de la imagen");
        }else {
            this.anchoMarco = ancho;
        }

    }


    /**
     * Establece la anchura del marco.
     * @param ancho  Ancho que se le va a dar al marco. Máximo: la mitad del menor valor entre el alto y el ancho de la imagen.
     */

    public void setAnchura(int ancho) {

        int maxAnchoMarco = 1/2*(ImagenProcesable.getAlto()/ImagenProcesable.getAncho());

        if(anchoMarco > maxAnchoMarco && anchoMarco < 0) {
            this.anchoMarco = 0;
            throw new IllegalArgumentException ("El parámetro ancho es menor que 0 o mayor que la mitad del menor valor entre el ancho y el alto de la imagen");
        }else {
            this.anchoMarco = ancho;
```

```
59          }
60      }
61
62
63
64      /**
65       * Devuelve la anchura del marco.
66       * @return Ancho para construir el marco.
67       */
68      public int getAnchura() {
69
70          return anchoMarco;
71      }
72
73      /**
74       * Establece el color del marco.
75       * @param color  Color para realizar el marco.
76       */
77      public void setColor(ColorRGB color) {
78
79          if(color != null) {
80              colorMarco = color;
81          }
82      }
83
84
85      /**
86       * Devuelve el color del marco.
87       * @return Color para realizar el marco.
88       */
89
90      public ColorRGB getColor() {
91          return colorMarco;
92      }
93
94
95      /**
96       * Aplica el efecto deseado: Introduce el marco parametrizado en la imagen. El
    resultado sobreescribe los píxeles necesarios de la imagen "para procesar" guardada en la
    superclase.
97       * Este método puede ser llamado varias veces. Por ejemplo: podríamos llamarlo una vez
    para realizar un marco azul de ancho 20 píxeles, cambiar el color a verde y el ancho a 10
    píxeles, y
98       * volver a llamar al método "aplicar". De esta forma obtendríamos un marco formado por
    2 colores (primero 10 píxeles en verde y después 10 píxeles en azul)
99       */
100
101     public void aplicar() {
102
103         int altoImagen = ImagenProcesable.getAlto();
104         int anchoImagen = ImagenProcesable.getAncho();
105         modificado = true;
106
107         // Pintar borde superior//
108         for(int y = 0; y < getAnchura(); y++) {
109             for(int x = 0; x < anchoImagen; x++) {
110
111                 ImagenProcesable.setPixel(y, x, colorMarco);
112             }
113         }
114
115         // Pintar borde inferior//
```

```
116
117         for(int y = altoImagen - getAnchura(); y < altoImagen ; y++) {
118             for(int x = 0; x < anchoImagen; x++) {
119
120                 ImagenProcesable.setPixel(y, x, colorMarco);
121             }
122         }
123
124         // Pintar borde izquierdo//
125
126         for(int x = 0; x < getAnchura(); x++) {
127             for(int y = getAnchura(); y < altoImagen - getAnchura(); y++)  {
128
129                 ImagenProcesable.setPixel(y,x, colorMarco);
130             }
131         }
132
133         // Pintar borde derecho//
134
135         for(int x = anchoImagen -getAnchura(); x < anchoImagen; x++) {
136             for(int y = getAnchura(); y < altoImagen - getAnchura(); y++)  {
137
138                 ImagenProcesable.setPixel(y, x,colorMarco);
139             }
140         }
141     }
142 }
143
144
145
146
147
148
149
150
151
152
```