Tarea 3
Proceso Digital de Imágenes

Ailyn Rebollar Pérez

La tarea se realizó en el lenguaje de programación Java en un proyecto de Netbeans así que para la ejecución del programa que muestra los filtros que se dejaron, basta con correr el proyecto Tarea3 que viene dentro de la carpeta comprimida llamada *Tarea3*.

Ahora bien, dentro del paquete tarea3 se encuentran 3 archivos .java los cuales son **FiltraImagenes.java**, **LetrasFiltros.java** y **JFrame.java**. En el primer archivo se sobreescribió el método **accept** que implementa FileFilter, esto con el propósito de que el programa únicamente pueda cargar imágenes con extensión *.jpg, .jpeg, .png y .gif* y el método **getDescription** para indicar al usuario las extensiones de las imágenes que se pueden cargar.

Dentro del archivo **LetrasFiltros** se creó ésta misma clase la cual sólo tiene como atributos el ancho y alto de las vecindades que se considerarán para el filtro mosaico y cada uno de los métodos que están ahí son los filtros que se dejaron de tarea implementar:

1. **m(BufferedImage imagen)**: El cual a partir de la imagen que se le pasa como parámetro crea un un archivo HTML donde imprime la imagen hecha con letras M con el color correspondiente en el archivo *m_color.html* .

2. **mGris(BufferedImage imagen)** El cual aplica primero el filtro de tonos de grises ocupando el promedio o media de los tres colores RGB y después construye la imagen en un archivo *m_gris.html*.

3. **letrasGris(BufferedImage imagen)**: Recibe una imagen y lo que hace es de acuerdo a su color en RGB se asigna una letra, por ejemplo si el tono es negro o muy cercano se escribe la M y si el tono es muy cercano al blando se deja la cadena vacía "" y regresa un archivo HTML llamado *letras_gris.html*.

4. **letrasColor(BufferedImage imagen)**: A la imagen que se recibe se le aplica el filtro mosaico y de acuerdo a su color promedio determinamos la letra a escribir, sin embargo respetamos el color original del pixel y se regresa en un archivo HTML *letras_color.html*.

5. **letras_tonosGris(BufferedImage imagen)**: Se aplica primero a la imagen recibida el filtro de tonos de Gris usando el promedio de los colores RGB y sucesivamente de acuerdo al tono se le asigna una letra para regresar al final un archivo HTML llamado *letras_tonosgris.html*.

6. **texto(BufferedImage imagen, String texto)**: Éste filtro toma el texto que el usuario ingreso y lo que hace es ir checando la longitud del texto elegido para asignar una letra a cada pixel respetando el color original regresandola en un archivo HTML llamado *texto.html*.

7. **domino(BufferedImage imagen)**: Aplica el filtro de tonos de grises sacando el promedio o media para después de acuerdo al tono asignar la ficha correspondiente considerando el lado izquierdo y derecho de la ficha en un archivo llamado *domino.html*.

8. **dominoNegro(BufferedImage imagen)**: Aplica el filtro de tonos de grises sacando el promedio o media para después de acuerdo al tono asignar la ficha correspondiente del domino negro considerando el lado izquierdo y derecho de la ficha en un archivo llamado *dominoNegro.html*.

9. **naipes(BufferedImage imagen)**: Aplica el filtro de tonos de grises sacando el promedio o media para después de acuerdo al tono asignar la carta correspondiente de los naipes en un archivo llamado *naipes.html*.