

Guía de mapas de bits con python

Importante: La presente guía se debe resolver en Python utilizando el módulo Pillow.

Ejercicio 1

Información general de una imagen

Implementar un programa en Python que haga lo siguiente:

- Solicite al usuario que ingrese la ruta de una imagen.
- Muestre la imagen en la pantalla.
- Retorne en formato tabla en la consola del sistema la siguiente información de la imagen:
 - Nombre
 - Extensión
 - Resolución
 - Cantidad de píxeles
 - Ubicación de la imagen (ruta)

Ejercicio 2

Guardando imágenes

Implementar un programa en Python que haga lo siguiente:

- Cargue una imagen desde un archivo.
- Muestre la imagen en la pantalla.
- Guarde una copia de la imagen con un nuevo nombre de manera persistente en el disco local del equipo.

Ejercicio 3

Rotando imagen

Implementar un programa en Python que haga lo siguiente:

- Solicite al usuario que ingrese la ruta de una imagen y el ángulo de rotación.
- Cargue dicha imagen dentro del script.
- Aplique la rotación deseada por el usuario.
- Muestre la imagen con la rotación aplicada en la pantalla.
- Guarde la imagen con la rotación aplicada de manera persistente con la siguiente nomenclatura *“nombreArchivoOriginalRot.extOriginal”*.

Ejercicio 4

Recortando imagen

Implementar un script en Python que solicite al usuario un nombre de imagen, una coordenada inicial, y los valores del ancho y altura que se pretenden recortar de la imagen original.

Posteriormente el script deberá guardar de manera persistente en el sistema el recorte de la imagen original en caso de que los parámetros suministrados sean válidos. En caso contrario se le debe indicar al usuario que los parámetros no lo son (por ejemplo si el nombre de la imagen a cargar no existe, o si la zona de recorte excede los límites de la imagen original).

Adicional: Se solicita guardar los recortes dentro de una carpeta “recortes” y con nomenclatura de nombres de archivos recorteX.png donde X representa el número de recortes realizados hasta la fecha.

Ejercicio 5

Marca de agua

Implementar un script en Python que solicite al usuario una ruta de la imagen que se desea marcar, posteriormente solicitar la ruta de la imagen de la marca de agua.

Luego el usuario deberá ingresar una de las siguientes cuatro opciones para determinar la ubicación de la marca de agua:

- Superior izquierda
- Superior derecha
- Inferior izquierda
- Inferior derecha

Finalmente el script deberá colocar la marca de agua en la zona especificada por el usuario manteniendo un margen de 50 px con los límites cercanos y descargar la imagen marcada de manera persistente en el sistema.

Ejercicio 6

Creando un collage

Implementar un script en Python solicite al usuario ingresar la ruta de una carpeta que contenga al menos 9 imágenes y la resolución del collage que se desea crear. Luego el script deberá alojar las rutas de cada imagen dentro de una lista, acto seguido se deberá implementar una función auxiliar ReSize que reciba una imagen en particular y retorne una miniatura de la misma.

Finalmente se debe construir un collage que incorpore todas las imágenes recibidas dentro de los límites de espacio que impone la resolución definida por el usuario al principio.

Importante: Se solicita que todo píxel del collage contenga información de alguna de las imágenes del usuario.

Ejercicio 7

Filtro blanco y negro

Implementar un script en Python solicite al usuario ingresar la ruta de una imagen. Luego el script deberá cargar la imagen en el Canvas y posteriormente se deberá implementar y aplicar un función que le aplique a la misma un filtro de blanco y negro.

Finalmente se debe guardar la imagen filtrada en la memoria local del sistema dentro de una carpeta de nombre “*ImagenesFiltradas*” con el nombre de archivo con la siguiente nomenclatura “*nombreOriginalBN.extOriginal*”.

Importante: El filtro se debe realizar de manera artesanal.

Ejercicio 8

Filtro gaussiano

Implementar un script en Python solicite al usuario ingresar la ruta de una imagen. Luego el script deberá cargar la imagen en el Canvas y posteriormente se deberá implementar y aplicar un función que le aplique a la misma un filtro gaussiano.

Finalmente se debe guardar la imagen filtrada en la memoria local del sistema dentro de una carpeta de nombre “*ImagenesFiltradas*” con el nombre de archivo con la siguiente nomenclatura “*nombreOriginalGaussian.extOriginal*”.

Importante: El filtro se debe realizar de manera artesanal.

Ejercicio 9

Filtro pixelator

Implementar un script en Python solicite al usuario ingresar la ruta de una imagen. Luego el script deberá cargar la imagen en el Canvas y posteriormente se deberá implementar y aplicar un función que le aplique a la misma un filtro pixelator.

Finalmente se debe guardar la imagen filtrada en la memoria local del sistema dentro de una carpeta de nombre “*ImagenesFiltradas*” con el nombre de archivo con la siguiente nomenclatura “*nombreOriginalPixelator.extOriginal*”.

Importante: El filtro se debe realizar de manera artesanal.