Para el programa que usamos para preprocesar las imágenes usamos las siguientes librerías

```
import os
import cv2
import numpy as np
from albumentations import (
    Compose, RandomBrightnessContrast, Rotate,
    RandomScale, GaussNoise, HorizontalFlip,
    Resize
)
from tqdm import tqdm
```

OS fue usada para creación del directorio de salida

Cv2 sirve para el preprocesamiento de imágenes

Albumentations fue usada para modificar las imágenes (como el añadir ruido)

Tqdm sirve para mostrar una barra de progreso al momento de procesar las imágenes

```
# Definir las transformaciones con albumentations
transform = Compose([
    RandomBrightnessContrast(brightness_limit=0.5, contrast_limit=0.5, p=1.0),
    Rotate(limit=30, p=1.0),
    RandomScale(scale_limit=0.2, p=1.0),
    GaussNoise(var_limit=(0, 0.05*255), p=0.3),
    HorizontalFlip(p=0.5),
])
```

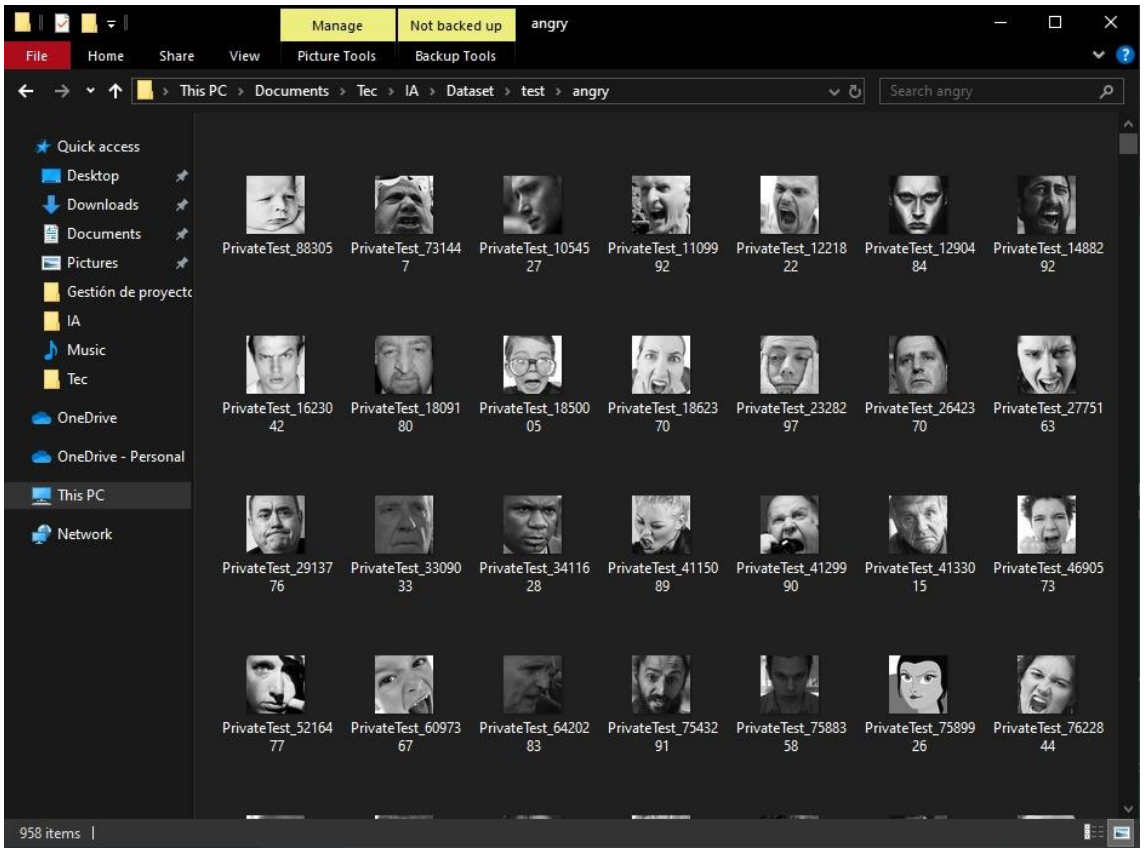
Aquí especificamos los parámetros que vamos a usar al momento de aumentar las imágenes.

```python
def process_and_save_image(image_path, output_path, augment_times=3):
    """
    Lee una imagen, aplica aumentos y guarda las versiones aumentadas
    :param image_path: path de la imagen original
    :param output_path: path donde guardar las imágenes aumentadas
    :param augment_times: cuántas versiones aumentadas crear (además de la original)
    """
    # Leer la imagen
    image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
    if image is None:
        print(f"Error al leer la imagen: {image_path}")
        return

    # Asegurarse de que es 48x48
    if image.shape != (48, 48):
        image = cv2.resize(image, (48, 48))

    # Guardar la imagen original primero
    base_name = os.path.splitext(os.path.basename(image_path))[0]
    cv2.imwrite(os.path.join(output_path, f"{base_name}_original.jpg"), image)

    # Crear versiones aumentadas
    for i in range(augment_times):
        augmented = transform(image=image)
        augmented_image = augmented['image']
        cv2.imwrite(os.path.join(output_path, f"{base_name}aug{i+1}.jpg"), augmented_image)
```

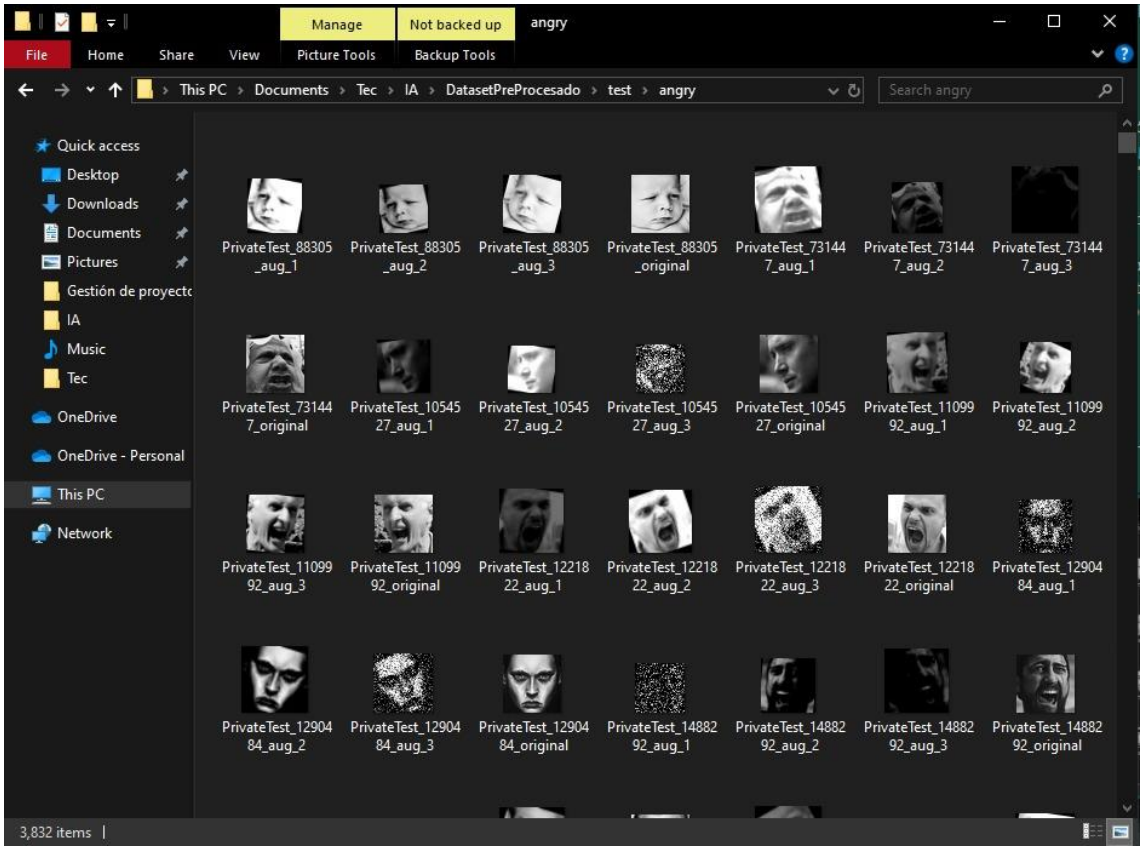
Aquí es donde se crean las imágenes, guarda el valor de las imágenes originales para después crear las imágenes aumentadas con los parámetros definidos previamente.

```python
def process_dataset(subset='train'):
    """
    Procesa todas las imágenes en el subset (train o test)
    :param subset: 'train' o 'test'
    """
    print(f"\nProcesando {subset}...")
    input_path = os.path.join(original_dataset_path, subset)
    output_path = os.path.join(output_dataset_path, subset)

    # Crear estructura de directorios
    if not os.path.exists(output_path):
        os.makedirs(output_path)

    for emotion in emotions:
        emotion_input_path = os.path.join(input_path, emotion)
        emotion_output_path = os.path.join(output_path, emotion)

        if not os.path.exists(emotion_output_path):
            os.makedirs(emotion_output_path)

        # Procesar cada imagen en la carpeta de emoción
        image_files = [f for f in os.listdir(emotion_input_path)
                       if f.lower().endswith(('.png', '.jpg', '.jpeg'))]

        print(f"Procesando {len(image_files)} imágenes para emoción: {emotion}")

        for image_file in tqdm(image_files, desc=emotion):
            image_path = os.path.join(emotion_input_path, image_file)
            process_and_save_image(image_path, emotion_output_path)

# Procesar ambos conjuntos (train y test)
process_dataset('train')
process_dataset('test')

print("\n¡Pre-procesamiento completado! Las imágenes aumentadas se han guardado en:", output_dataset_path)
```

Aquí se filtran por emociones las imágenes procesadas, y se guardan en su directorio correspondiente.



Imágenes sin aumentar



Imágenes aumentadas