Procesamiento de imágenes

1) Determina la matriz de salida al convolucionar la siguiente matriz de entrada:

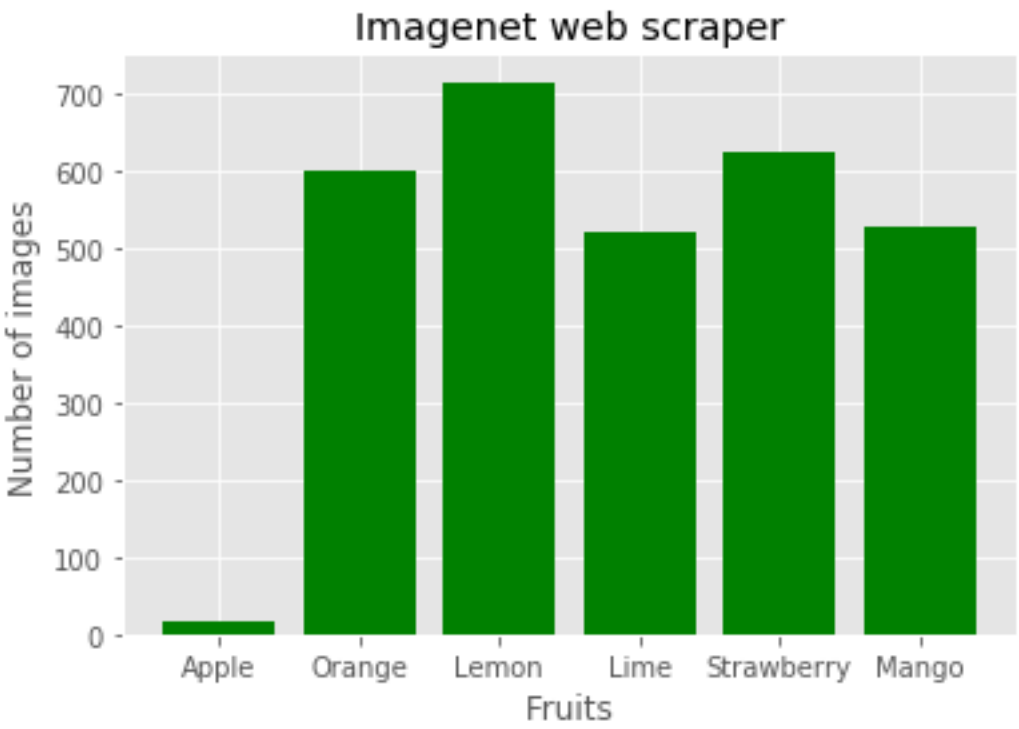
3	5	2	8	1
9	7	5	4	3
2	0	6	1	6
6	3	7	9	2

Con el siguiente kernel:

1	0	0
1	1	0
0	0	1

2) Cuando realizamos una convolución con una matriz de entrada A_{6x6} y un kernel C_{3x3}, obtuvimos una matriz de salida con dimensiones D_{4x4}. Dada una matriz de entrada A_{NxN} y un kernel C_{FxF} cuáles serían las dimensiones de nuestra matriz de salida D.

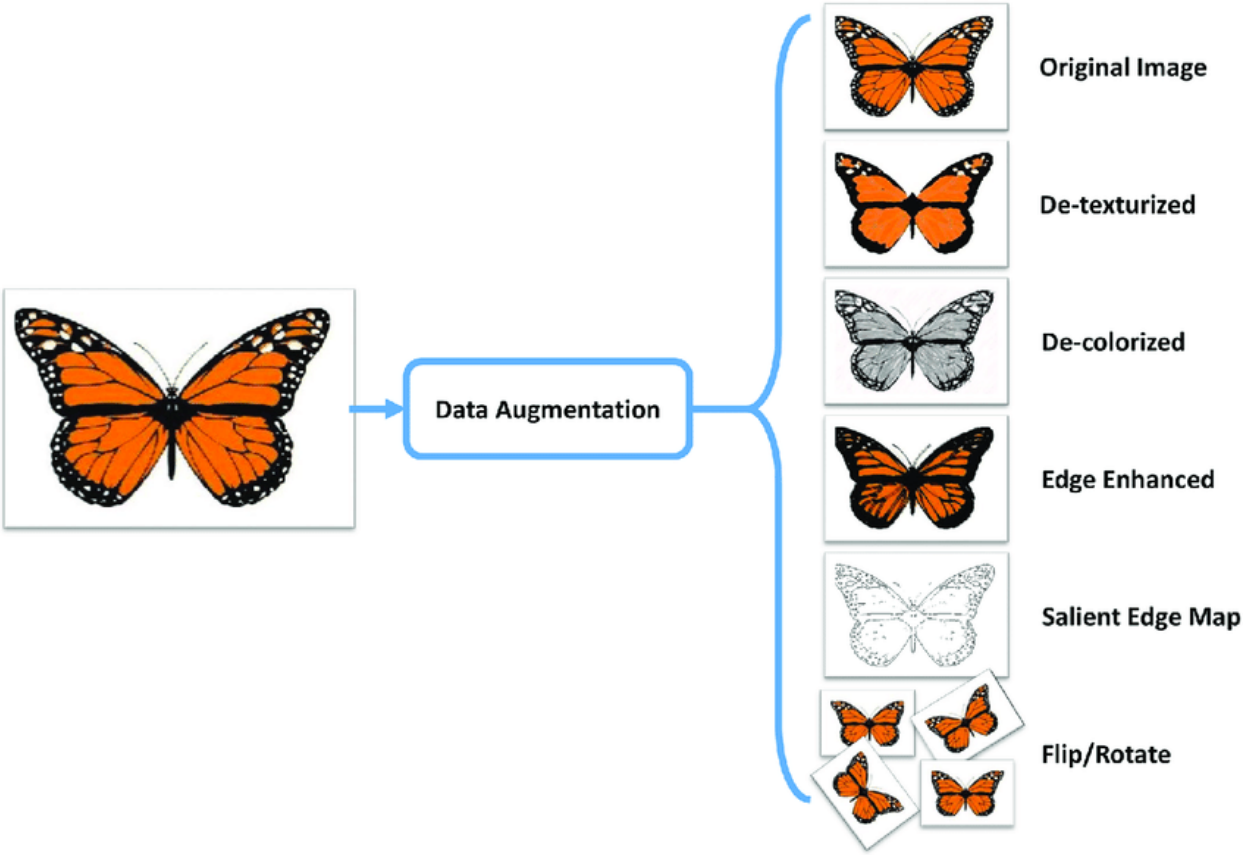
3) En la actividad integradora de este módulo diseñarán un *web scraper* para recolectar imágenes de diferentes objetos. Es importante visualizar cuántas imágenes el web scraper pudo recolectar, para identificar si hay un desbalance en la cantidad de imágenes que se recolectaron.



Al visualizar los datos un problema que podemos detectar es el desbalance de las clases, para arreglarlo se puede recolectar más datos de la minoría. No obstante, en ocasiones no es posible conseguir más datos por cuestiones de tiempo, costo o falta de equipo. En dichas instancias se puede hacer un submuestreo de la clase mayoritaria, esto implica descartar elementos de la categoría dominante al tiempo de entrenar el modelo. El problema con esta solución es que se desperdicia información que se recolectó. Otra aproximación es hacer un sobre muestreo de la clase minoritaria, lo cual implica duplicar elementos. El sobre muestreo no es una solución ideal, dado que su impacto se ve limitado porque los nuevos datos son redundantes. Una estrategia más favorable que el sub y sobre muestreo es *data augmentation*, el cual consta en generar datos artificiales modificando ligeramente los que ya se tienen a disposición. En el caso de imágenes, *data augmentation* se puede llevar a cabo mediante:

- Rotaciones
- Traslaciones
- Descolonizaciones

- Efecto espejo



Programen en Python un código que seleccione una imagen en su directorio llamada 'image_A.jpg' y haga *data augmentation*, las nuevas imágenes deben ser una rotación de 180° grados con respecto a la original y una nueva imagen con un efecto espejo, guarden las imágenes como 'rotate.jpg' y 'mirror.jpg'.

4) Programen en Python un código que seleccione una imagen en su directorio llamada 'image_B.jpg' y la convierta en escala de grises y haga que sus dimensiones de anchura (width) sean de 300 y altura (height) de 200. Guarden la imagen generada como 'resize.jpg'.

5) Efectúen el siguiente tutorial para extraer los momentos de hu de una imagen: https://www.learnopencv.com/shape-matching-using-hu-moments-c-python/