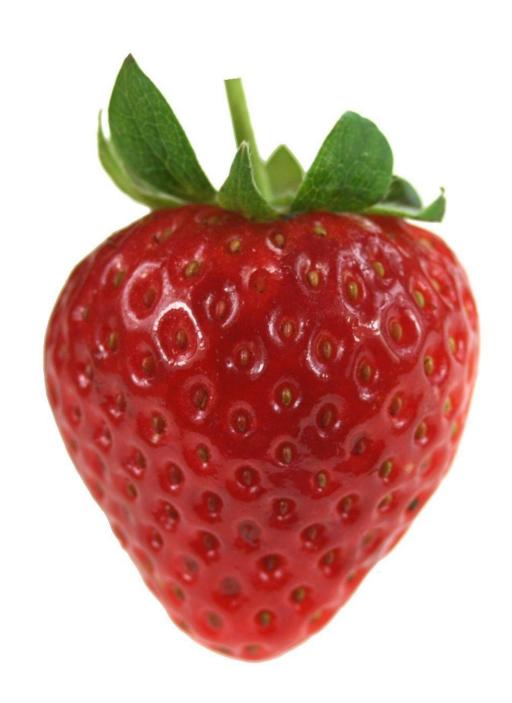
##Lab Exercise 2: Gaussian and Laplacian Pyramids • Objective: Learn how to build multi-resolution pyramids for an image.

• **Task:** Create Gaussian and Laplacian pyramids for a given image, then use these pyramids to perform image blending between two images.

```
import cv2 as cv
import numpy as np,sys
from google.colab.patches import cv2_imshow # Import the cv2_imshow
function

A = cv.imread('strawberry.jpg')
B = cv.imread('lychee.jpg')
assert A is not None, "file could not be read, check with
os.path.exists()"
assert B is not None, "file could not be read, check with
os.path.exists()"

A.shape,B.shape
((1283, 1200, 3), (900, 743, 3))
cv2_imshow(A)
cv2_imshow(B)
```





```
# Define the new dimensions (width, height)
new_width = 400
new_height = 400
```

```
# Resize the image
A resized = cv.resize(A, (new width, new height))
B_resized = cv.resize(B, (new_width, new_height))
# Save the resized image
cv.imwrite('Resized Apple.jpg', A resized)
cv.imwrite('Resized Orange.jpg', B resized)
image A = cv.imread('Resized Apple.jpg')
image B = cv.imread('Resized Orange.jpg')
image A.shape, image B.shape
((400, 400, 3), (400, 400, 3))
# generate Gaussian pyramid for A
G = image A.copy()
qpA = [G]
for i in range(6):
    G = cv.pyrDown(G)
    gpA.append(G)
# generate Gaussian pyramid for B
G = image B.copy()
gpB = [G]
for i in range(6):
    G = cv.pyrDown(G)
    qpB.append(G)
# generate Laplacian Pyramid for A
lpA = [gpA[5]]
for i in range(5,0,-1):
    GE = cv.pyrUp(qpA[i])
    # Resize GE to match the shape of gpA[i-1]
    GE = cv.resize(GE, (gpA[i-1].shape[1], gpA[i-1].shape[0]))
    L = cv.subtract(gpA[i-1],GE)
    lpA.append(L)
# generate Laplacian Pyramid for B
lpB = [qpB[5]]
for i in range(5,0,-1):
    GE = cv.pyrUp(qpB[i])
    # Resize GE to match the shape of gpB[i-1]
    GE = cv.resize(GE, (gpB[i-1].shape[1], gpB[i-1].shape[0]))
    L = cv.subtract(qpB[i-1],GE)
    lpB.append(L)
# Now add left and right halves of images in each level
LS = []
for la,lb in zip(lpA,lpB):
```

```
rows, cols, dpt = la.shape
    ls = np.hstack((la[:,0:cols//2], lb[:,cols//2:]))
    LS.append(ls)
# now reconstruct
ls = LS[0]
for i in range(1,6):
    ls = cv.pyrUp(ls)
    # Resize ls to match the shape of LS[i] before adding
    ls_ = cv.resize(ls_, (LS[i].shape[1], LS[i].shape[0]))
    ls = cv.add(ls , LS[i])
# image with direct connecting each half
real = np.hstack((image A[:,:cols//2],image B[:,cols//2:]))
cv.imwrite('Pyramid blending2.jpg',ls )
cv.imwrite('Direct blending.jpg',real)
True
# reading the images
Direct Blending = cv.imread('Direct blending.jpg')
Pyramid Blending= cv.imread('Pyramid blending.jpg')
# image with direct connecting each half
real = np.hstack((image A[:,:cols//2],image B[:,cols//2:]))
# Ensure the path is correct and the file is created successfully
cv.imwrite('/content/Pyramid blending2.jpg',ls )
cv.imwrite('/content/Direct blending.jpg',real)
# reading the images
# Update the paths to match the previous write operations
Direct Blending = cv.imread('/content/Direct blending.jpg')
Pyramid Blending= cv.imread('/content/Pyramid blending2.jpg')
cv2 imshow(Direct Blending)
cv2_imshow(Pyramid_Blending)
```

