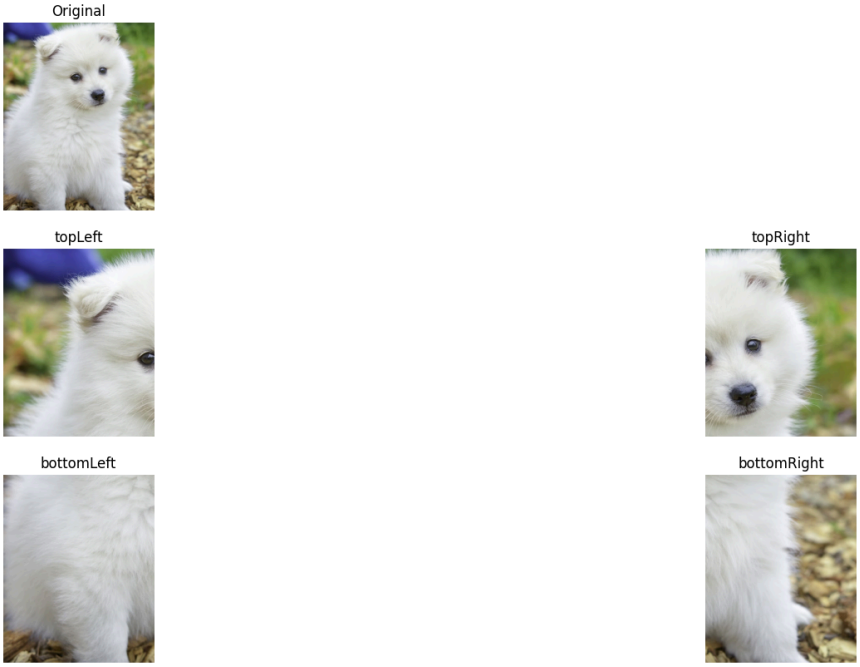
```
pip install opencv-python
```

```
Requirement already satisfied: opencv-python in /usr/local/lib/python3.10/dist-packages (4.8.0.76)
Requirement already satisfied: numpy>=1.21.2 in /usr/local/lib/python3.10/dist-packages (from opencv-python) (1.25.2)
```

```
#Program-7 Write a Program to read a digital image. Split and display image into 4 quadrants, up, down, right and left.
import cv2
from matplotlib import pyplot as plt
# Load image
image = cv2.imread('/content/puppy.jpg')
# create figure
fig = plt.figure(figsize=(20, 10))
# setting values to rows and column variables
rows = 3
columns = 2

#Get the height and width of the image
(h, w) = image.shape[:2]
#Converting BGR to RGB
img2=image[:,:,::-1]
#cv2.imshow('Original', image)
(cX, cY) = (w // 2, h // 2)
# crop the image into four parts which will be labelled as
# top left, top right, bottom left, and bottom right.
topLeft = image[0:cY, 0:cX]
img3=topLeft[:,:,::-1]
topRight = image[0:cY, cX:w]
img4=topRight[:,:,::-1]
bottomLeft = image[cY:h, 0:cX]
img5=bottomLeft[:,:,::-1]
bottomRight = image[cY:h, cX:w]
img6=bottomRight[:,:,::-1]

# Adds a subplot at the 1st position
fig.add_subplot(rows, columns, 1)
# showing image
plt.imshow(img2)
plt.axis('off')
plt.title("Original")

# Adds a subplot at the 2nd position
fig.add_subplot(rows, columns, 3)
# showing image
plt.imshow(img3)
plt.axis('off')
plt.title("topLeft")


# Adds a subplot at the 3rd position
fig.add_subplot(rows, columns, 4)
# showing image
plt.imshow(img4)
plt.axis('off')
plt.title("topRight")


# Adds a subplot at the 4th position
fig.add_subplot(rows, columns, 5)
# showing image
plt.imshow(img5)
plt.axis('off')
plt.title("bottomLeft")


# Adds a subplot at the 4th position
fig.add_subplot(rows, columns, 6)
# showing image
plt.imshow(img6)
plt.axis('off')
plt.title("bottomRight")
```

```
Text(0.5, 1.0, 'bottomRight')
```



```
# Program-8 program to show rotation, scaling, and translation on an image
#Python program to explain cv2.rotate() method, cv2.resize(),translate

# importing cv2
import cv2
from matplotlib import pyplot as plt
import numpy as np
fig = plt.figure(figsize=(20, 10))
rows = 3
columns = 2
# Reading an image in default mode
src = cv2.imread('/content/puppy.jpg')

# Window name in which image is displayed
window_name = 'Image'

# Using cv2.rotate() method
# Using cv2.ROTATE_90_CLOCKWISE rotate
# by 90 degrees clockwise
image = cv2.rotate(src, cv2.ROTATE_90_CLOCKWISE)

# Adds a subplot at the 1st position
fig.add_subplot(rows, columns, 1)

# showing image
img2=src[:,:,::-1]
plt.imshow(img2)
plt.axis('off')
plt.title("Original")

# Adds a subplot at the 2nd position
fig.add_subplot(rows, columns, 2)

# showing image
img3=image[:,:,::-1]
plt.imshow(img3)
plt.axis('off')
plt.title("Rotated")
# Displaying the image
img2=image[:,:,::-1]
plt.imshow(img2)
#cv2.imshow(window_name, image)
#cv2.waitKey(0)

r = 150.0 / image.shape[1]
dim = (150, int(image.shape[0] *r))
#r, c = src.shape
img_shrinked = cv2.resize(src, dim, interpolation=cv2.INTER_AREA)

# Adds a subplot at the 2nd position
fig.add_subplot(rows, columns, 3)

# showing image
img4=img_shrinked[:,:,::-1]
plt.imshow(img4)
plt.axis('off')
plt.title("scaled")

# shift the image 25 pixels to the right and 50 pixels down
M = np.float32([[1, 0, 25], [0, 1, 50]])
shifted = cv2.warpAffine(src, M, (src.shape[1], src.shape[0]))
# Adds a subplot at the 2nd position
fig.add_subplot(rows, columns, 4)

# showing image
img5=shifted[:,:,::-1]
plt.imshow(img5)
plt.axis('off')
plt.title("Translated")
```

```
Text(0.5, 1.0, 'Translated')
```


```
#Program-10 Write a program to blur and smoothing an image.
#Smoothing an image using an average blur.
#Notice as how the kernel size increases, the image becomes progressively more blurred.
# importing cv2
import cv2
from matplotlib import pyplot as plt
import numpy as np
fig = plt.figure(figsize=(20, 10))
rows = 3
columns = 2
# Reading an image in default mode
src = cv2.imread('D:\\puppy.jpg')
i=1
kernelSizes = [(3, 3), (9, 9), (15, 15)]
# loop over the kernel sizes
for (kX, kY) in kernelSizes:
    # apply an "average" blur to the image using the current kernel
    # size
    blurred = cv2.blur(src, (kX, kY))
    fig.add_subplot(rows, columns, i)
    #cv2.imshow("Average ({}, {})".format(kX, kY), blurred)
    #cv2.waitKey(0)


    # showing image
    img4= blurred[:,:,::-1]
    plt.imshow(img4)
    plt.axis('off')
    plt.title("blurred")
    i+=1
```

```
#Program-11 Write a program to contour an image.
# Grayscale
import cv2
from matplotlib import pyplot as plt
import numpy as np
fig = plt.figure(figsize=(20, 10))
rows = 3
columns = 2
# Reading an image in default mode
src = cv2.imread('D:\\puppy.jpg')
gray = cv2.cvtColor(src, cv2.COLOR_BGR2GRAY)

# Find Canny edges
edged = cv2.Canny(gray, 30, 200)

contours, hierarchy = cv2.findContours(edged,cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)

fig.add_subplot(rows, columns, 1)


plt.imshow(edged)
plt.axis('off')
plt.title("Canny Edges After Contouring")

print("Number of Contours found = " + str(len(contours)))

# Draw all contours
# -1 signifies drawing all contours
cv2.drawContours(src, contours, -1, (0, 255, 0), 3)


fig.add_subplot(rows, columns, 2)

# showing image
img11= src[:,:,::-1]
plt.imshow(img11)
plt.axis('off')
plt.title("Contours")
```

```
Number of Contours found = 44
Text(0.5, 1.0, 'Contours')
```


```
#Program-12 Write a program to detect a face/s in an image
#https://www.datacamp.com/tutorial/face-detection-python-opencv
import cv2
from matplotlib import pyplot as plt
import numpy as np

# Reading an image in default mode
src = cv2.imread('D:\\pic1.jpg')
gray_image = cv2.cvtColor(src, cv2.COLOR_BGR2GRAY)
face_classifier = cv2.CascadeClassifier(
    cv2.data.haarcascades + "haarcascade_frontalface_default.xml"
)
face = face_classifier.detectMultiScale(
    gray_image, scaleFactor=1.1, minNeighbors=5, minSize=(40, 40)
)
for (x, y, w, h) in face:
    cv2.rectangle(src, (x, y), (x + w, y + h), (0, 255, 0), 4)
img_rgb = cv2.cvtColor(src, cv2.COLOR_BGR2RGB)
```