7) Write a Program to read a digital image. Split and display image into 4 quadrants, up, down, right and left.

```
import cv2
image = cv2.imread('ivan.jpg')
height,width,_ = image.shape
halfh= height//2
halfw= width//2
top_left = image[:halfh,:halfw]
top_right = image[:halfh,halfw:]
bottom_left = image[halfh:,:halfw]
bottom_right = image[halfh:,halfw:]
cv2.imshow("top_left",top_right)
cv2.imshow("top_right",top_right)
cv2.imshow("bottom_left",bottom_left)
cv2.imshow("bottom_right",bottom_right)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

8) program to show rotation, scaling, and translation on an image.

```
import cv2
import numpy as np
image = cv2.imread('face.jpg')
height,width,_ = image.shape
rotation_matrix = cv2.getRotationMatrix2D((width/2,height/2), 45, 1)
scale_matrix = np.float32([[1.5,0,0],[0,1.5,0]])
translation_matrix = np.float32([[1,0,100],[0,1,50]])
rotate = cv2.warpAffine(image, rotation_matrix, (width,height))
scale = cv2.warpAffine(image, scale_matrix,(int(width*1.5),int(height*1.5)))
translate = cv2.warpAffine(image, translation_matrix, (width,height))
cv2.imshow("original",image)
cv2.imshow("rotate",rotate)
cv2.imshow("scale",scale)
cv2.imshow("transate",translate)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

9) Read an image and extract and display low-level features such as edges, textures using filtering techniques.

```
import cv2
import numpy as np
image_path = "face.jpg"  # Replace with your image path
img = cv2.imread(image_path)
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
laplacian_edges = cv2.Laplacian(img, cv2.CV_64F)
laplacian_edges = cv2.normalize(laplacian_edges, None, 0, 255,
cv2.NORM_MINMAX,dtype=cv2.CV_8U)
edges = cv2.Canny(gray, 100, 200)
kernel = np.ones((5, 5), np.float32) / 25
texture = cv2.filter2D(gray, -1, kernel)
cv2.imshow("Original Image", img)
cv2.imshow("Edges (Laplacian Filter)", laplacian_edges)
cv2.imshow("Edges (Canny Detector)", edges)
```

```
cv2.imshow("Texture", texture)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

10) Write a program to blur and smoothing an image.

```
import cv2
image=cv2.imread('org.jpg')
gaussian=cv2.GaussianBlur(image,(5,5),0)
median=cv2.medianBlur(image,5)
bilayteral=cv2.bilateralFilter(image,9,75,75)
cv2.imshow("original",image)
cv2.imshow("Gaussian",gaussian)
cv2.imshow('median',median)
cv2.imshow("filter",bilayteral)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

11) Write a program to contour an image.

```
import cv2
image=cv2.imread("dar.jpg")
gray=cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
_,thresh=cv2.threshold(gray,0,255,cv2.THRESH_BINARY_INV)
contours,_=cv2.findContours(thresh,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
contour=image.copy()
cv2.drawContours(contour,contours,-1,(0,255,0),2)
cv2.imshow("original",image)
cv2.imshow("contur",contour)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

12) Write a program to detect a face/s in an image.

```
import cv2
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
image = cv2.imread('sop.jpg')
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5,
minSize=(30, 30))
for (x, y, w, h) in faces:
    cv2.rectangle(image, (x, y), (x+w, y+h), (255, 0, 0), 2)
cv2.imshow('Faces Detected', image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```