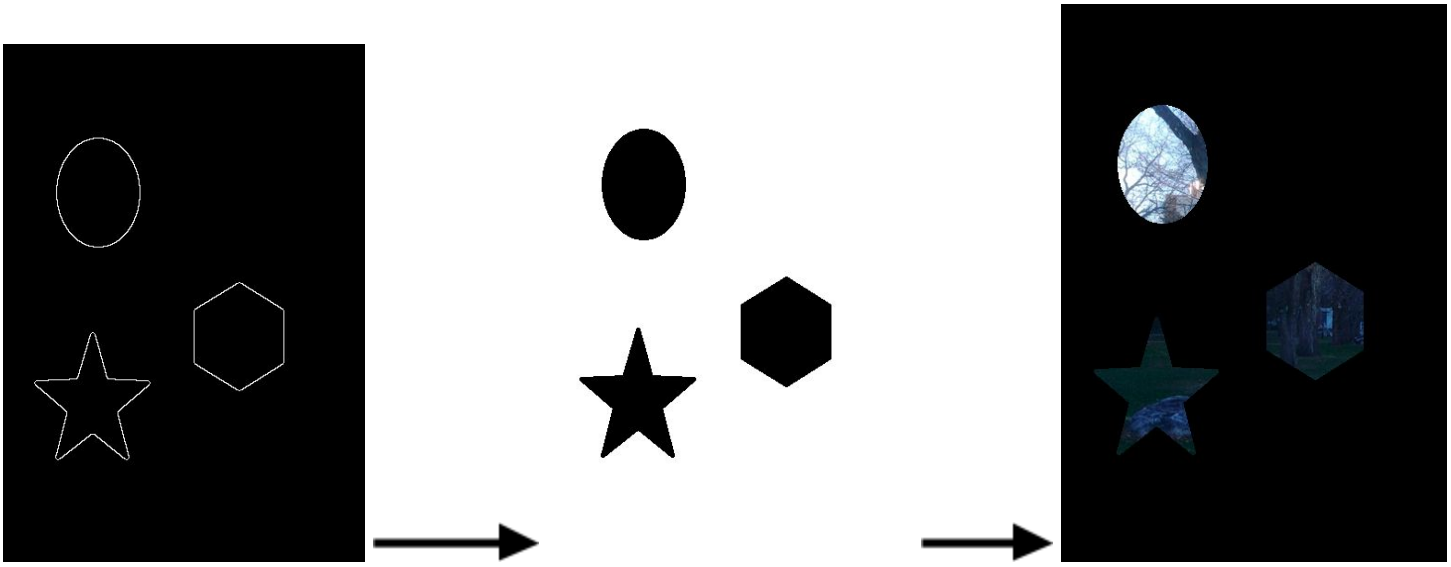
ASSIGNMENT - 1
MM - 805

DAMAYANTI GHOSH
ID : 1505758

➢ The workflow followed in achieving the result in this assignment is as follows :



➢ The above has been coded in OpenCV Python and is as follows

```
#importing the necessary libraries
import numpy as np
import cv2
import imutils

#loading the images
image1 = cv2.imread('campus.png')
image = cv2.imread('maskedge.png')


#preprocessing the image for thresholding for contour detection
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

blurred = cv2.GaussianBlur(gray,(5,5), 0)
thresh = cv2.threshold(blurred, 0, 255, cv2.THRESH_BINARY)[1]

#contour detection
cnts = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

cnts = cnts[0] if imutils.is_cv2() else cnts[1]
print(len(cnts)) #3 - the number of contours

#drawing the contours in the format of 'maskbw.png'
cv2.drawContours(image, cnts, -1, (255,255,255), -1)
image = cv2.bitwise_not(image)

#making a copy of the image1
result = image1

#finding the dimension of the image
h, w, d = image.shape

#for the corresponding white spaces(pixel position-wise) transforming the exact pixel(position-wise) of the  image to black
for i in range(0, h):
    for j in range(0, w):
        for k in range(0,d):
            if(image[i,j,k]==255):
                result[i,j,k]=0

#loading the resultant image
cv2.imshow("Result", result)
cv2.waitKey(0)
```

➢ **Brief Description of the Main Idea**

The first image ‘maskedge.png’ has been preprocessed and made to undergo thresholding for contour detection. The resultant image - ‘image’ is then made to undergo “bitwise-not” to resemble the image “maskbw.png”. Meanwhile , a copy of the image “campus.png” , ‘image1’ is made and for every pixel location of the ‘image’ where it is black, the exact same pixels (at the exact pixel-wise position) of ‘image1’ (the copy of ‘campus.png’) are made black as well to achieve the target image.