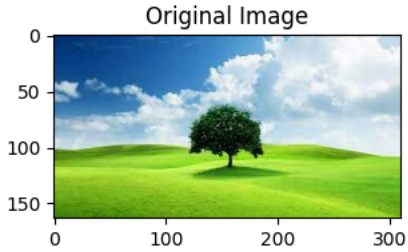
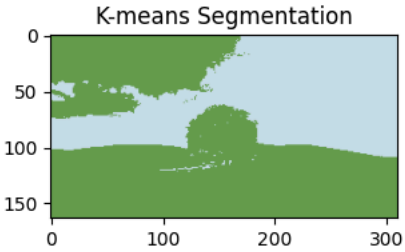
```
from google.colab import files
import os
import cv2
import numpy as np
from skimage.filters import threshold_otsu
import matplotlib.pyplot as plt

# Define a function to upload multiple images
def upload_images(image_filenames):
    uploaded_files = files.upload()
    for image_filename in image_filenames:
        if image_filename in uploaded_files:  # Check if the file is uploaded
            try:
                with open(image_filename, 'wb') as f:
                    f.write(uploaded_files[image_filename])
                print(f"{image_filename} uploaded successfully!")
            except Exception as e:
                print(f"Error uploading {image_filename}: {e}")
        else:
            print(f"File not found: {image_filename}")

# List of image file names
image_filenames = ["Img1.jpeg", "Img2.jpeg","Img3.jpeg","Img4.jpeg"]  # Add your image file names here

# Upload the images
upload_images(image_filenames)

# Process uploaded images
for filename in image_filenames:
    img = cv2.imread(filename)

    if img is None:
        print(f"Error: Unable to load image {filename}")
        continue

    ##Plot Original Image
    plt.figure(figsize=(10, 6))
    plt.subplot(3, 3, 1)
    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.title('Original Image')

    ##Preprocessing the Image
    img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
    twoDimage = img.reshape((-1,3))
    twoDimage = np.float32(twoDimage)

    ##Defining Parameters
    criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 10, 1.0)
    K = 2
    attempts=10

    ##Apply K-Means
    ret,label,center=cv2.kmeans(twoDimage,K,None,criteria,attempts,cv2.KMEANS_PP_CENTERS)
    center = np.uint8(center)
    res = center[label.flatten()]
    result_image = res.reshape((img.shape))

    ##Plot K-means Segmentation Image
    plt.subplot(3, 3, 2)
    plt.imshow(result_image)
    plt.title('K-means Segmentation')

    ##Plot K-means Classifying
    plt.subplot(3, 3, 3)
    plt.imshow(label.reshape(img.shape[:2]), cmap='gray')
    plt.title('K-means Classifying')

    #Image Segmentation using Contour Detection
    img = cv2.resize(img,(256,256))

    gray = cv2.cvtColor(img,cv2.COLOR_RGB2GRAY)
    _,thresh = cv2.threshold(gray, np.mean(gray), 255, cv2.THRESH_BINARY_INV)
    edges = cv2.dilate(cv2.Canny(thresh,0,255),None)

    ##Detecting and Drawing Contours

    cnt = sorted(cv2.findContours(edges, cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)[-2], key=cv2.contourArea)[-1]
    mask = np.zeros((256,256), np.uint8)
    masked = cv2.drawContours(mask, [cnt],-1, 255, -1)
```

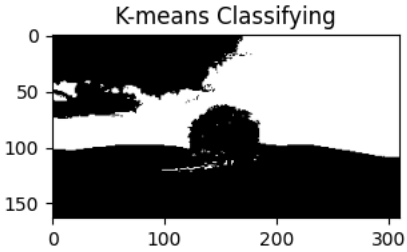
```
##Plot Contour Detection
plt.subplot(3, 3, 4)
plt.imshow(masked, cmap='gray')
plt.title('Contour Detection')

##Segmenting the Regions

dst = cv2.bitwise_and(img, img, mask=mask)
segmented = cv2.cvtColor(dst, cv2.COLOR_BGR2RGB)

#Plot Segmented Regions
plt.subplot(3, 3, 5)
plt.imshow(segmented)
plt.title('Segmented Regions')

#Image Segmentation using Thresholding

##Preprocessing the Image
img_rgb=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
img_gray=cv2.cvtColor(img_rgb,cv2.COLOR_RGB2GRAY)

##Segmentation Process
def filter_image(image, mask):
    r = image[:,:,0] * mask
    g = image[:,:,1] * mask
    b = image[:,:,2] * mask
    return np.dstack([r,g,b])

thresh = threshold_otsu(img_gray)
img_otsu  = img_gray < thresh
filtered = filter_image(img, img_otsu)

##Plot Thresholding
plt.subplot(3, 3, 6)
plt.imshow(filtered)
plt.title('Thresholding')

#Segmentation using Color Masking
## Preprocessing the Image

rgb_img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
hsv_img = cv2.cvtColor(rgb_img, cv2.COLOR_RGB2HSV)

##Define the Color Range to be Detected
light_blue = (90, 70, 50)
dark_blue = (128, 255, 255)
# You can use the following values for green
# light_green = (40, 40, 40)
# dark_greek = (70, 255, 255)
mask = cv2.inRange(hsv_img, light_blue, dark_blue)

##Apply the Mask
result = cv2.bitwise_and(img, img, mask=mask)

##Plot Color Masking
plt.subplot(3, 3, 7)
plt.imshow(result)
plt.title('Color Masking')

plt.tight_layout()
plt.show()
```

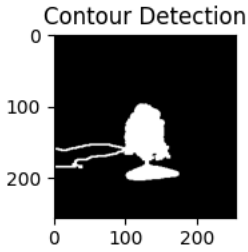
Choose Files No file chosen

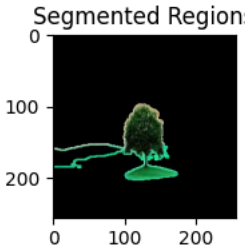
```
File not found: Img1.jpeg
File not found: Img2.jpeg
File not found: Img3.jpeg
File not found: Img4.jpeg
```

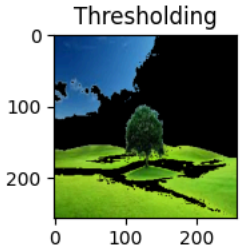


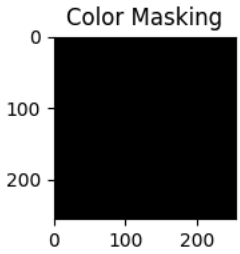


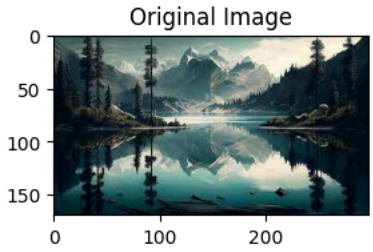


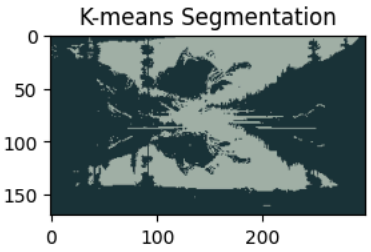


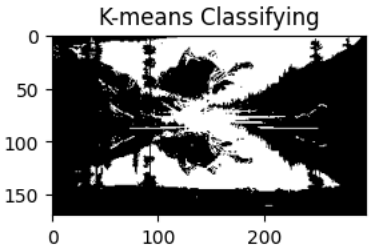


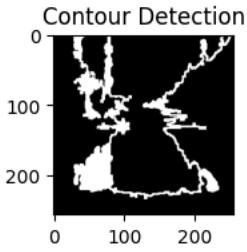


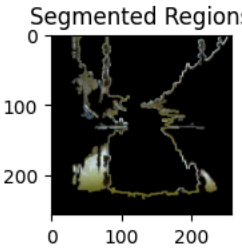


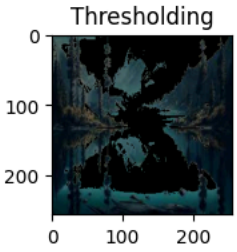


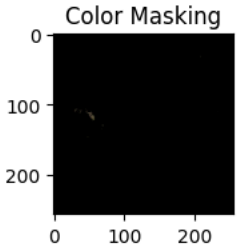


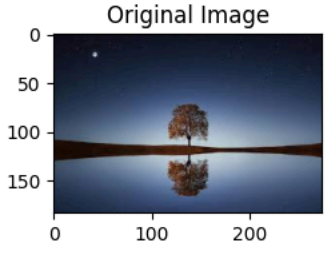


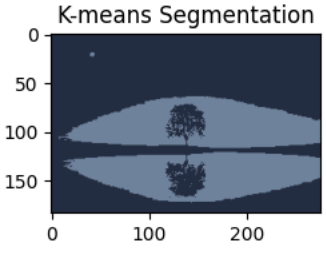


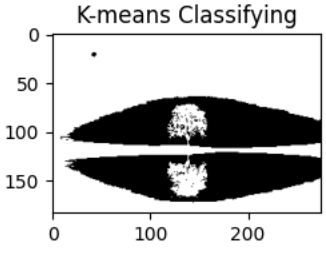


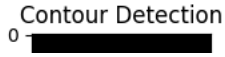


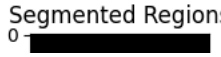


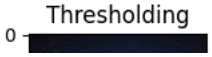


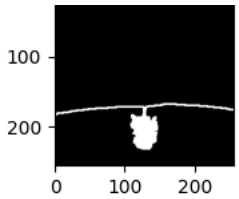


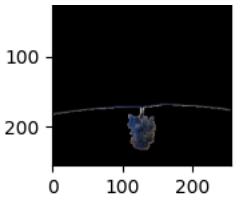


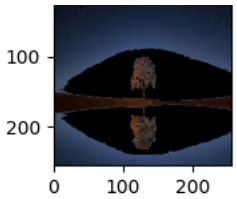


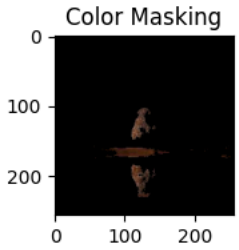


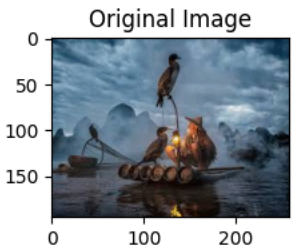


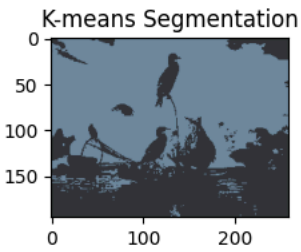

100
200
0
100
200

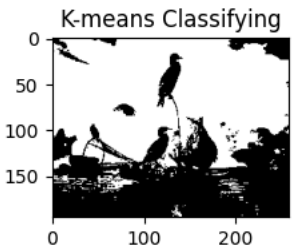

100
200
0
100
200

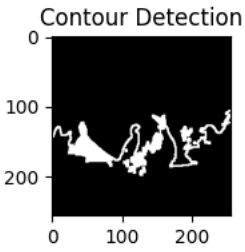

100
200
0
100
200

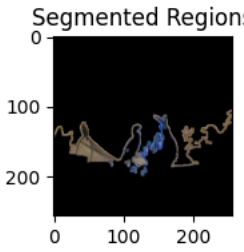

Color Masking
0
100
200
0
100
200

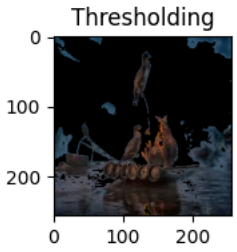

Original Image
0
50
100
150
0
100
200

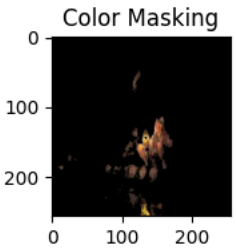

K-means Segmentation
0
50
100
150
0
100
200


K-means Classifying
0
50
100
150
0
100
200


Contour Detection
0
100
200
0
100
200


Segmented Regions
0
100
200
0
100
200


Thresholding
0
100
200
0
100
200


Color Masking
0
100
200
0
100
200