```
#Algorithm
#Take the query image and convert it to grayscale.
#Now Initialize the ORB detector and detect the keypoints in query image and scene.
#Compute the descriptors belonging to both the images.
#Match the keypoints using Brute Force Matcher.
#Show the matched images.


import numpy as np
import cv2
from google.colab.patches import cv2_imshow

# Read the query image as query_img
# and traing image This query image
# is what you need to find in train image
# Save it in the same directory
# with the name image.jpg
query_img = cv2.imread('/content/drive/MyDrive/Images_for_ORB/query_img_bw.jpg')
train_img = cv2.imread('/content/drive/MyDrive/Images_for_ORB/train_image_bw.jpg')


# Convert it to grayscale
query_img_bw = cv2.cvtColor(query_img,cv2.COLOR_BGR2GRAY)
train_img_bw = cv2.cvtColor(train_img, cv2.COLOR_BGR2GRAY)

# Initialize the ORB detector algorithm
orb = cv2.ORB_create()



# Now detect the keypoints and compute
# the descriptors for the query image
# and train image
queryKeypoints, queryDescriptors = orb.detectAndCompute(query_img_bw,None)
trainKeypoints, trainDescriptors = orb.detectAndCompute(train_img_bw,None)


# Initialize the Matcher for matching
# the keypoints and then match the
# keypoints
matcher = cv2.BFMatcher()
matches = matcher.match(queryDescriptors,trainDescriptors)


# draw the matches to the final image
# containing both the images the drawMatches()
# function takes both images and keypoints
# and outputs the matched query image with
# its train image
final_img = cv2.drawMatches(query_img, queryKeypoints,
train_img, trainKeypoints, matches[:20],None)
```

```
final_img = cv2.resize(final_img, (1000,650))


# Show the final image
cv2_imshow(final_img)
cv2.waitKey(3000)
```


```
-1
```