I. Level 1 – Rotate image

- Đọc ảnh vào từ file thông qua lệnh imread và điều chỉnh kích thước ảnh theo tỷ lệ thông qua hàm resize.

```
def resize(image, scale = 0.75):
    width = int(image.shape[1] * scale)
    height = int(image.shape[0] * scale)
    dimensions = (width,height)

    return cv.resize(image, dimensions, interpolation=cv.INTER_CUBIC)

# read the image
image = cv.imread('lv1/lv1_1.jpg')
image = resize(image,0.5)
original = image.copy()
```

- Làm mờ ảnh, chuyển ảnh qua grayscale và threshold nhằm giảm bớt số lượng contour tìm được.
    - Dùng RETR_EXTERNAL để chỉ tìm contour bao phủ bên ngoài của ảnh nhằm giảm bớt số contour không cần thiết.

```
# find the contours
blur = cv.GaussianBlur(image,(3,3),0)
gray = cv.cvtColor(image,cv.COLOR_BGR2GRAY)
thresh = cv.threshold(gray, 0, 255, cv.THRESH_BINARY|cv.THRESH_OTSU)[1]
contours, hierarchy =cv.findContours(thresh, cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE)
```

- Tạo ra 2 điểm khác biệt:
    - Tạo biến n để lưu vị trí contour hiện tại.
    - Tạo 2 biến dif1 và dif2 là vị trí contour bị thay đổi.
    - For tất cả các contour, nếu như là contour bị thay đổi:
        - Tạo khung thay đổi
        - Cắt ảnh từ khung thay đổi trong ảnh gốc lưu vào biến test và thay khung thay đổi trong ảnh gốc bằng màu nền.
        - Xoay ảnh thông qua getRotationMatrix2D và warpAffine.
        - Đặt lại ảnh được xoay vào ảnh gốc
    - Trong tất cả các trường hợp, biến n đều tăng nhằm theo dõi vị trí hiện tại

```
# make the different
n=0
dif1 = random.randint(0,len(contours))
dif2 = random.randint(0,len(contours))

for contour in contours:
    if(n==dif1 or n==dif2):
        n = n+1
        x, y, w, h = cv.boundingRect(contour)
        test = image[y:y+h,x:x+w].copy()
        image[y:y+h,x:x+w] = (191,146,112)
        height, width = test.shape[:2]
        rotation_matrix = cv.getRotationMatrix2D((width/2,height/2),random.randint(1,3)*90, 1)
        rotated_test = cv.warpAffine(test,rotation_matrix,(width,height))
        image[y:y+h,x:x+w] = rotated_test
    n=n+1
```

- Sửa lại background trong trường hợp cắt ảnh sót lại.

```
# correct background
mask = cv.inRange(image,(0,0,0),(100,100,100))
image[mask>0] = (191,146,112)
```

- Lưu ảnh vào file.

```
# save image to file
cv.imwrite('lv1/lv1_2.jpg',image)
```

- Tạo khoảng trống và ghép ảnh được chỉnh sửa và ảnh gốc vào 2 bên của khoảng trống.
- In ảnh.

```
# output image
blank = np.zeros((original.shape[0],30,3),dtype='uint8')
final = cv.hconcat([image, blank, original])
cv.imshow('Level 1',final)

cv.waitKey(0)
cv.destroyAllWindows()
```

II. Level 2 – Change color

- Đọc ảnh vào từ file thông qua lệnh imread và điều chỉnh kích thước ảnh theo tỷ lệ thông qua hàm resize.

```
def resize(image, scale = 0.75):
    width = int(image.shape[1] * scale)
    height = int(image.shape[0] * scale)
    dimensions = (width,height)

    return cv.resize(image, dimensions, interpolation=cv.INTER_CUBIC)

# read the image
image = cv.imread('lv2/lv2_1.jpg')
image = resize(image,0.5)
original = image.copy()
```

- Làm mờ ảnh, chuyển ảnh qua grayscale và threshold nhằm giảm bớt số lượng contour tìm được.
    - Dùng RETR_EXTERNAL để chỉ tìm contour bao phủ bên ngoài của ảnh nhằm giảm bớt số contour không cần thiết.

```
# find contours
blur = cv.GaussianBlur(image,(3,3),0)
gray = cv.cvtColor(image,cv.COLOR_BGR2GRAY)
thresh = cv.threshold(gray, 0, 255, cv.THRESH_BINARY|cv.THRESH_OTSU)[1]
contours, hierarchy =cv.findContours(thresh, cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE)
```

- Tạo 4 điểm khác biệt:
    - Tạo biến n để lưu vị trí contour hiện tại.
    - Tạo 4 biến dif1, dif2, dif3, dif4 là vị trí contour bị thay đổi.
    - For tất cả các contour, nếu như là contour bị thay đổi:
        - Tạo khung thay đổi
        - Cắt ảnh từ khung thay đổi trong ảnh gốc lưu vào biến test.
        - Tìm trong ảnh được cắt màu hiện tại của hình mũi tên.
        - Thay đổi màu hình mũi tên thành màu ngẫu nhiên
        - Đặt lại ảnh được đổi màu vào ảnh gốc
    - Trong tất cả các trường hợp, biến n đều tăng nhằm theo dõi vị trí hiện tại

```
n=0
dif1 = random.randint(0,len(contours))
dif2 = random.randint(0,len(contours))
dif3 = random.randint(0,len(contours))
dif4 = random.randint(0,len(contours))

for contour in contours:
    if(n==dif1 or n==dif2 or n==dif3 or n==dif4):
        x, y, w, h = cv.boundingRect(contour)
        test = image[y:y+h,x:x+w].copy()
        mask_red = cv.inRange(test,(36-36,27-27,237-50),(36+50,27+50,255))
        mask_yellow = cv.inRange(test,(0,242-50,254-50),(0+50,255,255))
        mask_pink = cv.inRange(test,(201-50,174-50,254-50),(201+50,174+50,254+1))
        mask_green = cv.inRange(test,(77-50,177-50,35-35),(77+50,177+50,35+50))
        mask_orange = cv.inRange(test,(38-38,127-50,255-50),(38+50,127+50,255))
        mask_blue = cv.inRange(test,(232-50,163-50,0),(255,163+50,0+50))
        test[mask_red>0] = (random.randint(0,255),random.randint(0,255),random.randint(0,255))
        test[mask_yellow>0] = (random.randint(0,255),random.randint(0,255),random.randint(0,255))
        test[mask_pink>0] = (random.randint(0,255),random.randint(0,255),random.randint(0,255))
        test[mask_green>0] = (random.randint(0,255),random.randint(0,255),random.randint(0,255))
        test[mask_orange>0] = (random.randint(0,255),random.randint(0,255),random.randint(0,255))
        test[mask_blue>0] = (random.randint(0,255),random.randint(0,255),random.randint(0,255))
        image[y:y+h,x:x+w] = test
    n=n+1
```

- Lưu ảnh vào file.

```
# save image
cv.imwrite('lv2/lv2_2.jpg',image)
```

- Tạo khoảng trống và ghép ảnh được chỉnh sửa và ảnh gốc vào 2 bên của khoảng trống.
- In ảnh.

```
# output image
blank = np.zeros((original.shape[0],30,3),dtype='uint8')
final = cv.hconcat([image, blank, original])
cv.imshow('Level 2',final)

cv.waitKey(0)
cv.destroyAllWindows()
```

III. Different finder

1. Cách sử dụng

- Thay đổi lv trong ngoặc thành level muốn tìm điểm khác biệt và chạy chương trình, chương trình sẽ cho ra kết quả của level đó.

```
# select the level to find diff
lvl = 'lv2'
```

2. Chi tiết

- Tạo biến lưu trữ level tìm điểm khác biệt

```
# select the level to find diff
lvl = 'lv2'
```

- Đọc ảnh vào từ file thông qua lệnh imread và điều chỉnh kích thước ảnh theo tỷ lệ thông qua hàm resize.

```
def resize(frame, scale = 0.75):
    width = int(frame.shape[1] * scale)
    height = int(frame.shape[0] * scale)
    dimensions = (width,height)

    return cv.resize(frame, dimensions, interpolation=cv.INTER_CUBIC)

# load and resize the image
img1 = cv.imread(lvl+'/'+lvl+'_1.jpg')
img1 = resize(img1, 0.5)
img2 = cv.imread(lvl+'/'+lvl+'_2.jpg')
```

- Chuyển đổi 2 ảnh cần so sánh qua Grayscale, dùng hàm cv2.absdiff để tìm điểm khác nhau giữa 2 ảnh Grayscale.

```
# convert to grayscale and find diff
img1Gray = cv.cvtColor(img1, cv.COLOR_BGR2GRAY)
img2Gray = cv.cvtColor(img2, cv.COLOR_BGR2GRAY)
diff = cv.absdiff(img1Gray,img2Gray)
```

- Làm mờ ảnh, giãn ảnh và threshold để giảm bớt số contour không cần thiết.
    - Sử dụng hàm imutils.grab_contours để giảm bớt số lượng contour cần xét.

```
# find contours
threshold = cv.threshold(diff,0,255,cv.THRESH_BINARY|cv.THRESH_OTSU)[1]
kernel = np.ones((3,3),np.uint8)
dilate = cv.dilate(threshold, kernel, iterations=2)
contours = cv.findContours(dilate.copy(), cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE)
contours = imutils.grab_contours(contours)
```

- For tất cả các contour:
    - Xét những contour có diện tích lớn hơn 10 nhằm giảm số contour cần phải xét.
    - Tạo khung khác biệt và in hình chữ nhật ở vị trí khác nhau lên mỗi ảnh

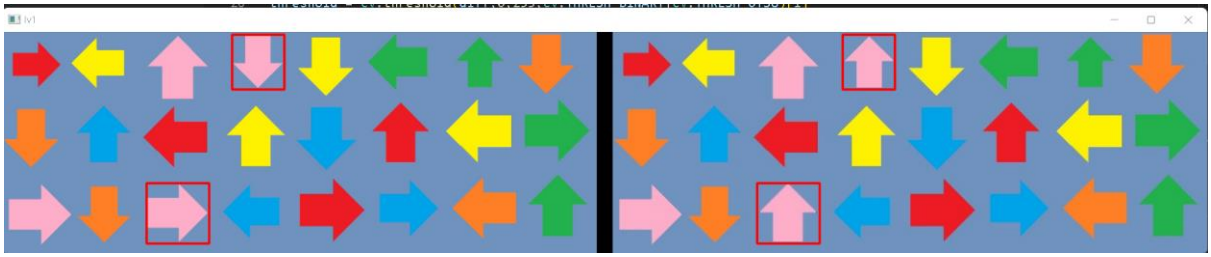
```
# loop over each contours to draw a box
for contour in contours:
    if(cv.contourArea(contour) > 10):
        x, y, w, h = cv.boundingRect(contour)
        cv.rectangle(img1, (x,y), (x+w,y+h), (0,0,255), 2)
        cv.rectangle(img2, (x,y), (x+w,y+h), (0,0,255), 2)
```

- Tạo khoảng trống và ghép ảnh được chỉnh sửa và ảnh gốc vào 2 bên của khoảng trống.
- In ảnh.

```
# final img
blank = np.zeros((img1.shape[0],20,3),dtype='uint8')
final = cv.hconcat([img1, blank, img2])
cv.imshow('Different',final)
```

3. Demo

a. Level 1



b. Level 2