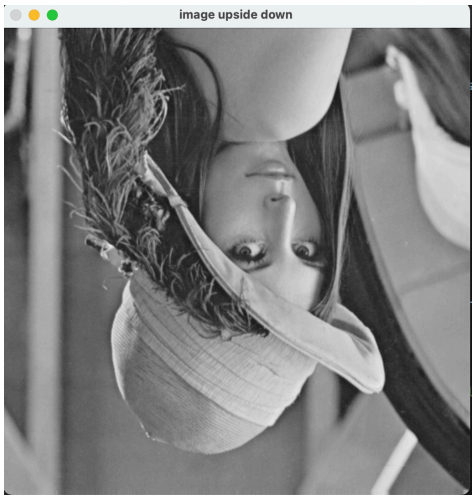
電腦視覺 HW1

R11944045 網媒所碩一 呂佳晏

Part 1.

(a) upside-down

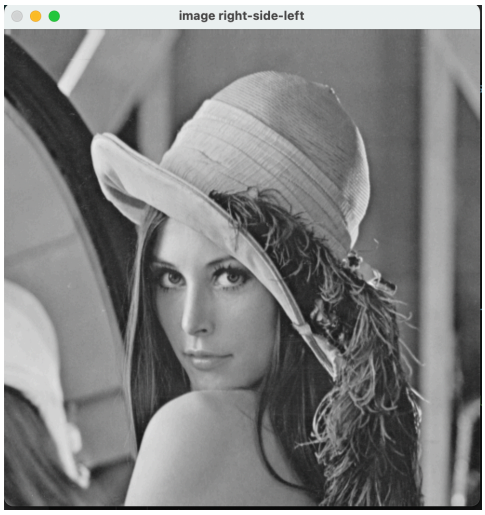
```python
#part 1.(a) image upside down
def image_upside_down(image_to_process):
    row = image_to_process.shape[0] #image 高度
    half = row // 2
    for i in range(half):
        image_to_process[[i]], image_to_process[[row - i - 1]] = image_to_process[[row - i - 1]], image_to_process[[i]]

    return image_to_process
```



我的想法是將上半部的像素和下半部的像素兩兩交換

(b) Right-side-left

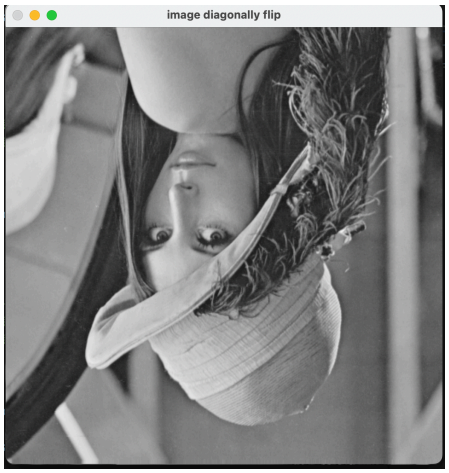
```python
#part 1.(b) image right-side-left
def image_right_side_left(image_to_process):
    row = image_to_process.shape[0] #image 高度
    col = image_to_process.shape[1] #image 寬度
    half = col // 2
    for i in range(row):
        for j in range(half):
            image_to_process[i][[j]], image_to_process[i][[col - 1 - j]] = image_to_process[i][[col - 1 - j]], image_to_process[i][[j]]
    return image_to_process
```



想法雷同 upside-down ， right-side-left只是改成將左半部的像素和右半部的像素兩兩交換

(c) Diagonally flip

```python
#part 1.(c) diagonally flip
def image_diagonally_flip(image_to_process):
    #do right-side-left first and than upside down the image
    row = image_to_process.shape[0] #image 高度
    col = image_to_process.shape[1] #image 寬度
    half = row // 2
    for i in range(row):
        for j in range(half):
            image_to_process[i][[j]], image_to_process[i][[col - 1 - j]] = image_to_process[i][[col - 1 - j]], image_to_process[i][[j]]
    for i in range(half):
        image_to_process[[i]], image_to_process[[row - i - 1]] = image_to_process[[row - i - 1]], image_to_process[[i]]
    return image_to_process
```

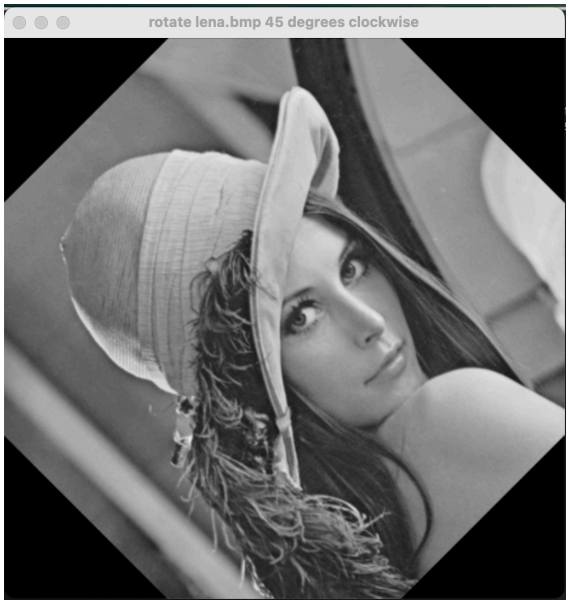


合併 upside-down 和 right-side-left，Diagonally flip 的想法是先將圖片左右翻轉後再上下翻轉

Part 2.

(d) rotate lena.bmp 45 degrees clockwise

```
#part 2.(d) rotate lena.bmp 45 degrees clockwise
image1 = cv.imread('lena.bmp')
row = image1.shape[0]
col = image1.shape[1]
center = (row / 2, col / 2)
matrix = cv.getRotationMatrix2D(center, 45, 1)
image_rotated = cv.warpAffine(image1, matrix, (row, col))
cv.imshow('rotate lena.bmp 45 degrees clockwise', image_rotated)
cv.waitKey(0)
```



我利用了 cv2 的 getRotationMatrix2D 函式，先得到旋轉矩陣，再利用 wrapAffine 這個函式對每個像素進行旋轉

(e) shrink lena.bmp in half

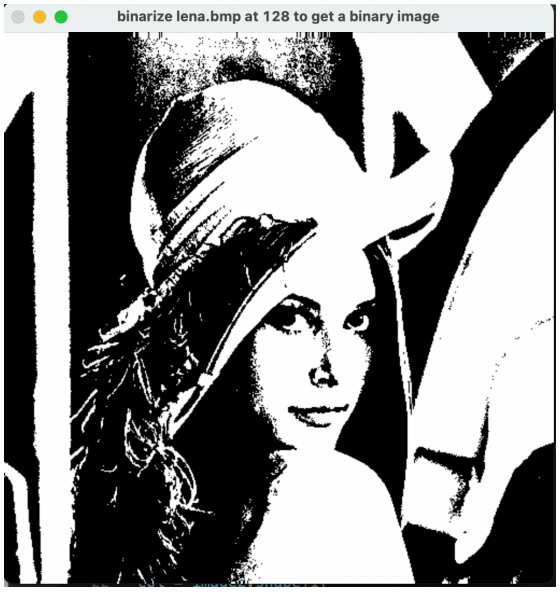
```
#part 2.(e) shrink lena.bmp in half
image2 = cv.imread('lena.bmp')
row = image2.shape[0]
col = image2.shape[1]
image_shrink = cv.resize(image, (row // 2, col // 2))
cv.imshow('shrink lena.bmp in half', image_shrink)
cv.waitKey(0)
```

我利用 cv2 內建的 resize 函式，將長寬都縮小為原來的一半

結果如上圖所示，跟原圖比起來長寬都是原本的一半

(f) binarize lena.bmp at 128 to get a binary image

```
#part 2.(f) binarize lena.bmp at 128 to get a binary image
image3 = cv.imread('lena.bmp')
retVal, image_binarize = cv.threshold(image3, 127, 255, cv.THRESH_BINARY)
cv.imshow('binarize lena.bmp at 128 to get a binary image', image_binarize)
cv.waitKey(0)
```



我利用 cv2 內建的 threshold 函式，將每個 digit 值 > 127 的點都設成白色 (255)，低於的設成黑色