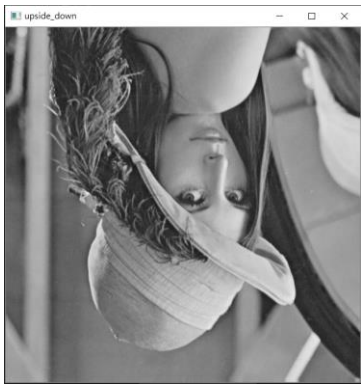
Part1

(a) upside-down

定義一個函式，將原圖以灰階開啟後，使用迴圈，將範圍限制在 0 至原圖 row 的大小，將第一列的所有 pixel 與最後一列互換，第二列與倒數第二列互換……直到最後一列，即完成圖片上下翻轉。



(b) right-side-left

定義一個函式，使用迴圈，將範圍限制在 0 至原圖 column 的大小，將第一行的所有 pixel 與最後一行互換，第二行與倒數第二行互換……直到最後一行，即完成圖片左右翻轉。

(c) diagonally flip

定義一個函式，使用雙層迴圈，第一層範圍限制在 0 至原圖 row 的大小，第二層範圍限制在 0 至原圖 column 的大小，在雙迴圈內，將原圖的橫向 pixel 從第一列，第二列……依序換到最後一列，倒數第二列……；縱向 pixel 從最後一行，倒數第二行……依序換到第一行，到二行……，即完成對角線翻轉。

Part2

(d) rotate 45 degrees clockwise

這題處理的方式使用 rotate 函數，rotate(image, angle, center, scale)，需要的 input 有要旋轉的角度、圖片中心和調整大小倍率，此題角度要順時針 45 度，就要輸入-45，圖片中心就是 height 和 width 的一半，大小不需改變，在來使用 M = cv.getRotationMatrix2D(center, angle, scale)與 cv.warpAffine(image, M, (w, h))即完成。

(e) shrink in half

此題使用 resize 函數，題目要求將原圖大小處理成原本的一半，所以 scale-percent = 50，將 width 和 height 乘上 scale-percent/100 得到 dim(width, height)，最後使用 cv.resize(img, dim, interpolation = cv.INTER_AREA)即可完成。

(f) binarize at 128 to get binary image

此題要將原圖做二值化分析，設定閥值在 128，使用函數 ret,binary =

cv.threshold(img, 127, 255, cv.THRESH_BINARY)，即可完成。

Source code:

```
import numpy as np
import cv2 as cv
from pylab import *

img = cv.imread('lena.bmp',0)
row = img.shape[0]
col = img.shape[1]
result = np.zeros([row,col],np.uint8)

#part1_a
def upside_down(img):
	for i in range(0, row, 1):
		result[i,:] = img[row-1-i,:]
	return result

p1_a = upside_down(img)
cv.imshow('upside_down', p1_a)

#part1_b
def right_side_left(img):
	for i in range(0, col, 1):
		result[:,i] = img[:,col-1-i]
	return result

p1_b = right_side_left(img)
cv.imshow('right_side_left', p1_b)
```

```
#part1_c
def diagonally_flip(img):
    for i in range(0,row,1):
        for j in range(0,col,1):
            result[row-i-1, j] = img[i, col-j-1]
    return result

p1_c = diagonally_flip(img)
cv.imshow('diagonally_flip', p1_c)

#part2_d
def rotate(image, angle, center = None, scale=1.0):
    (h, w) = img.shape[:2]
    if center is None:
        center = (w/2, h/2)

    M = cv.getRotationMatrix2D(center, angle, scale)
    rotated = cv.warpAffine(image, M, (w, h))
    return rotated

p2_d = rotate(img, -45)
cv.imshow('rotate', p2_d)

#part2_e
def shrink(img):
    scale_percent = 50 # percent of original size
    width = int(img.shape[1] * scale_percent / 100)
    height = int(img.shape[0] * scale_percent / 100)
    dim = (width, height)
    # resize image
    resized = cv.resize(img, dim, interpolation = cv.INTER_AREA)
    return resized

p2_e = shrink(img)
cv.imshow('shrink', p2_e)



#part2_f
```

```
def binarize(img):
    ret,binary = cv.threshold(img,127,255,cv.THRESH_BINARY)
    return binary

p2_f = binarize(img)
cv.imshow('binarize', p2_f)

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