# Computer Vision - Assignment 1

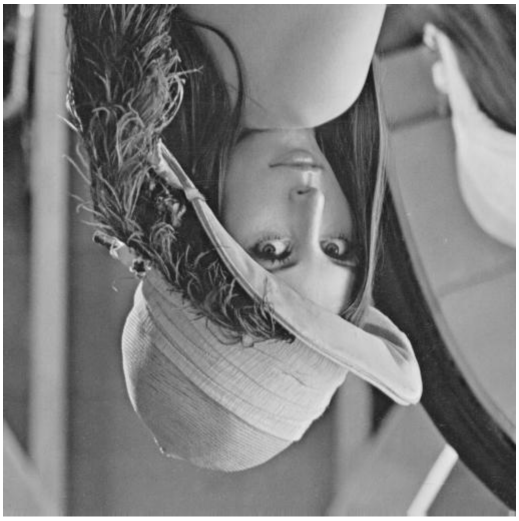
R09922A04 資工所人工智慧組 黃品硯

## Part 1

### (a) upside-down lena.bmp

Iteratively swap the first and last row of the image.

**[Output]**



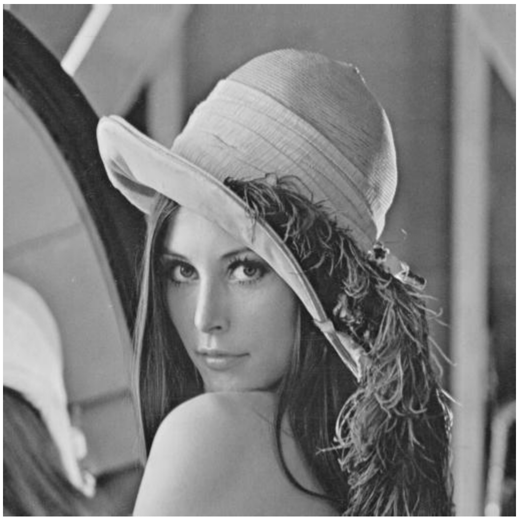
**[Code]**

```
for y in range(height // 2):
    rev_y = height - y - 1
    img[y, :, :], img[rev_y, :, :] = img[rev_y, :, :], img[y, :, :].copy()
```

### (b) right-side-left lena.bmp

Iteratively swap the first and last column of the image.

**[Output]**



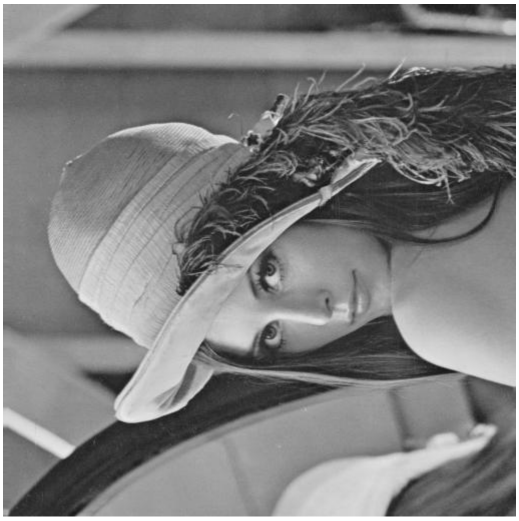
**[Code]**

```
for x in range(width // 2):
    rev_x = width - x - 1
    img[:, x, :], img[:, rev_x, :] = img[:, rev_x, :], img[:, x, :].copy()
```

## (c) diagonally flip lena.bmp

Iteratively swap the (x, y) pixel with (y, x) pixel.

**[Output]**



**[Code]**

```
for y in range(height):
    for x in range(y):
        img[y, x, :], img[x, y, :] = img[x, y, :], img[y, x, :].copy()
```

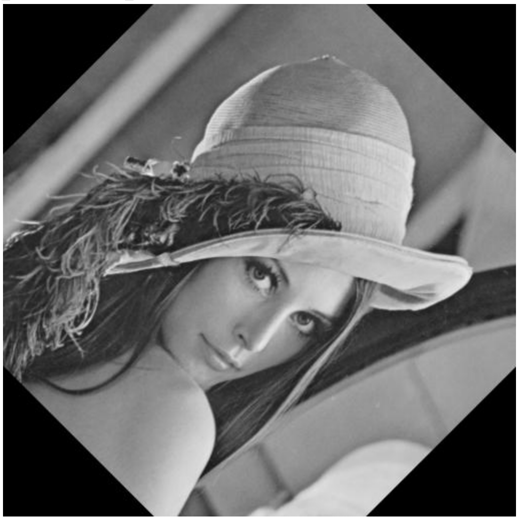
# Part 2

## (d) rotate lena.bmp 45 degrees clockwise

Get rotation matrix by OpenCV function, **cv2.getRotationMatrix2D.**

Use the rotation matrix to project the original image to new space.

**[Output]**



**[Code]**

```
def rotate(img, degree=-45):
    height, width, depth = img.shape
    img_center = (height / 2, width / 2)

    rot_mat = cv2.getRotationMatrix2D(img_center, degree, 1.0)
    result = cv2.warpAffine(img, rot_mat, (height, width))
```

## (e) shrink lena.bmp in half

Use the OpenCV function, **cv2.resize** to shrink the image in half.

**[Output]**

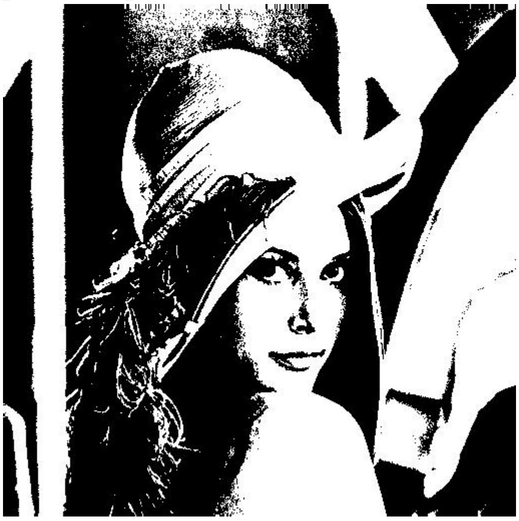
**[Code]**

```
def shrink(img, ratio=0.5):
    height, weight, depth = img.shape
    dim = (int(height * ratio), int(weight * ratio))
    result = cv2.resize(img, dim, cv2.INTER_AREA)
```

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

Binarize the image by OpenCV function, **cv2.threshold.**

**[Output]**



**[Code]**

```
def binarize(img, threshold=128):
    ret, result = cv2.threshold(img, threshold, 255, cv2.THRESH_BINARY)
```