Computer Vision Homework 5

Name: 黃新予
Student ID: f08922136

I. **Environment Setup**

Language: Python 3 (on VS code)

Library: numpy, PIL

II. **Q1: Gray scale dilation**

step 1: open lena.bmp as original image

step 2: create a new image as output image

step 3: traverse all pixels in original image. Overlap kernel on each pixel. Find the **maximum** value in the range that kernel value = 1.

step 4: Because K(x) = 0, we use local maximum as new pixel and set new pixel value to output image.

```python
def gs_Dilation(image, kernel, kernelCenter):
    '''
    gs = gray scale
    image: Image (from PIL)
    kernel: np.array
    kernelCenter: 2D tuple
    '''
    width, height = image.size
    gs_DilationImage = Image.new('L', image.size)
    for c in range(width):
        for r in range(height):
            newPixel = image.getpixel((c, r))
            for x in range(-kernelCenter[0], kernelCenter[0] + 1):
                for y in range(-kernelCenter[0], kernelCenter[0] + 1):
                    if (0 <= c+x < width and 0 <= r+y < height and kernel[kernelCenter[0]+x, kernelCenter[1]+y] == 1):
                        newPixel = max(newPixel, image.getpixel((c+x, r+y)))
            gs_DilationImage.putpixel((c, r), newPixel)

    return gs_DilationImage
```

III. **Q2: Gray scale erosion**

step 1: open lena.bmp as original image

step 2: create a new image as output image

step 3: traverse all pixels in original image. Overlap kernel on each pixel. Find the **minimum** value in the range that kernel value = 1.

step 4: Because K(x) = 0, we use local minimum as new pixel and set new pixel value to output image.

```python
def gs_Erosion(image, kernel, kernelCenter):
    '''
    gs = gray scale
    image: Image (from PIL)
    kernel: np.array
    kernelCenter: 2D tuple
    '''
    width, height = image.size
    gs_ErosionImage = Image.new('L', image.size)
    for c in range(width):
        for r in range(height):
            newPixel = image.getpixel((c, r))
            for x in range(-kernelCenter[0], kernelCenter[0] + 1):
                for y in range(-kernelCenter[0], kernelCenter[0] + 1):
                    if (0 <= c+x < width and 0 <= r+y < height and kernel[kernelCenter[0]+x, kernelCenter[1]+y] == 1):
                        newPixel = min(newPixel, image.getpixel((c+x, r+y)))
            gs_ErosionImage.putpixel((c, r), newPixel)

    return gs_ErosionImage
```

IV. **Q3: Gray scale opening**

step 1: call function gs_erosion first, then call function gs_dilation

```
def gs_Opening(image, kernel, kernelCenter):
    return gs_Dilation(gs_Erosion(image, kernel, kernelCenter), kernel, kernelCenter)
```

V. **Q4: Gray scale closing**

step 1: call function gs_dilation first, then call function gs_ erosion

```
def gs_Closing(image, kernel, kernelCenter):
    return gs_Erosion(gs_Dilation(image, kernel, kernelCenter), kernel, kernelCenter)
```