

背景：

使用 sobel operation，並且在過程中不能 call function，最後比

較原圖以及做完之後的圖片效果。

在此作業，我們需要實作 sobel operation。

實作方法

首先我們先把圖片輸入，並把其轉為 array 的形式。

```
image1 = Image.open('baboon.png').convert('L')
gray_image1 = np.array(image1)
```

將該 array 丟入 function 做處理

我們使用的 sobel operation 使用了兩個矩陣做 convolution

X 方向：

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * A$$

Y 方向

$$G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * A$$

```
sobel_x = np.array([[ -1,  0,  1], [ -2,  0,  2], [ -1,  0,  1]])
sobel_y = np.array([[ -1, -2, -1], [ 0,  0,  0], [ 1,  2,  1]])
```

在實作上，我們圖片中的每一個點會藉由 3\*3 的矩陣與對應的區塊相乘比全部加再一起得到 Gx 與 Gy 值

```
image_block = padded_image[x_start:x_end, y_start:y_end]

gradient_x = np.sum(np.multiply(image_block, sobel_x))
gradient_y = np.sum(np.multiply(image_block, sobel_y))
```

而當位置沒有值得時候，會藉由 padding 補值

```
padded_image = np.pad(image, pad_width=1, mode='constant', constant_values=0)
```

最後 gradient 的值為：

$$G = \sqrt{G_x^2 + G_y^2}$$

```
gradient = np.sqrt(gradient_x**2 + gradient_y**2)

new_image[i, j] = gradient
```

把 gradient 值限定在 0~255 之間





```
new_image *= 255.0 / new_image.max()
```

把 array 變回 image，並輸出結果。

```
sobel_image1 = Image.fromarray(sobel_image1)
sobel_image1.show()
```

如果助教需要實際操作一次，可以按照我 readme 的方式操作。

實驗結果:

	原圖	Sobel 後結果
baboon		
peppers		
pool	