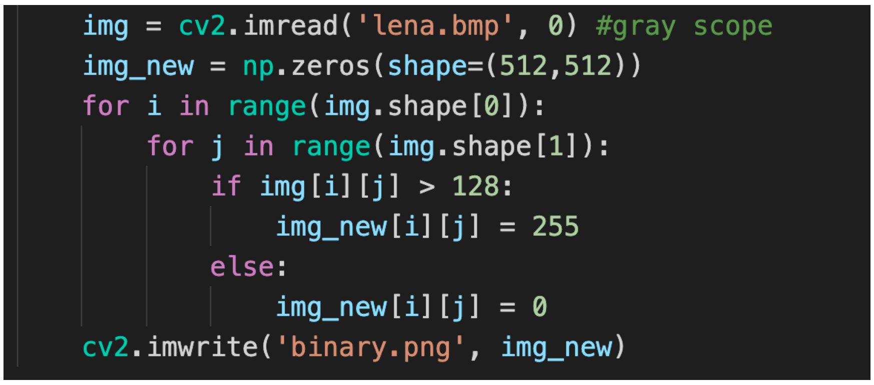
HW2
r10944013 網媒所碩一 馮啟倫

HW2-1:
目標：
將照片轉換為 binary image (threshold at 128)

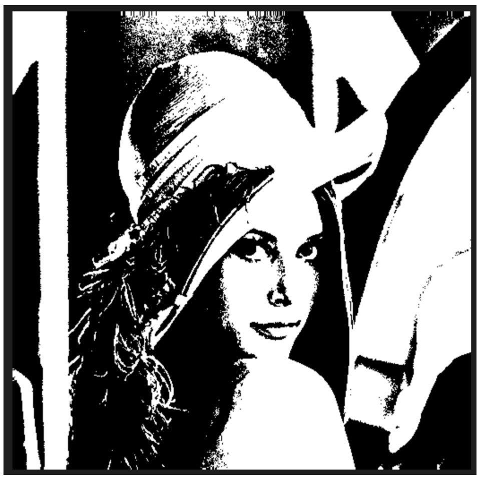
演算法：
我將512*512個像素點全部掃過一次，只要大於128就設定為255, 反之則設為0

Code segment:



```
img = cv2.imread('lena.bmp', 0) #gray scope
img_new = np.zeros(shape=(512,512))
for i in range(img.shape[0]):
    for j in range(img.shape[1]):
        if img[i][j] > 128:
            img_new[i][j] = 255
        else:
            img_new[i][j] = 0
cv2.imwrite('binary.png', img_new)
```

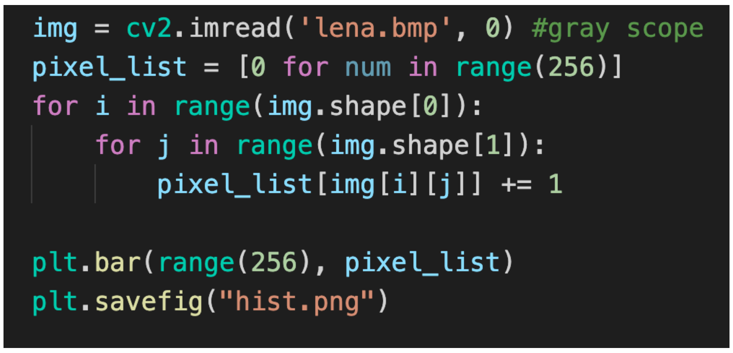
Result image:



HW2-2:
目標：將照片的像素轉為histogram

演算法：
我先把整個照片做像素的數值統計，得到一個256長度的list，然後根據這個list，使用plt.bar繪製histogram

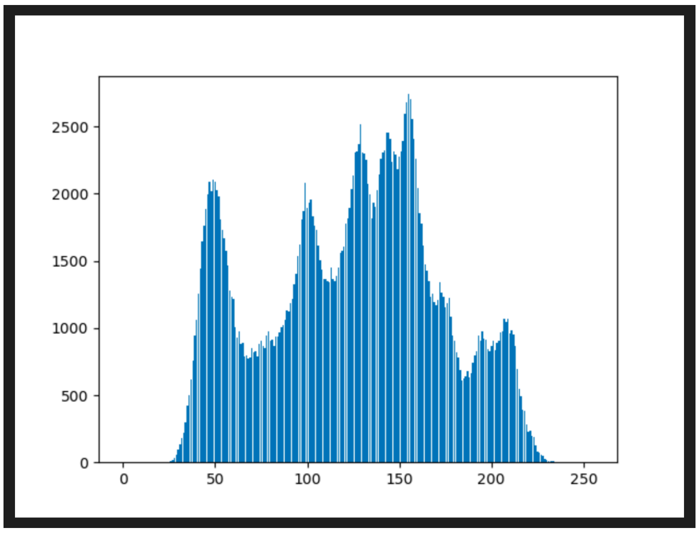
Code segment



```python
img = cv2.imread('lena.bmp', 0) #gray scope
pixel_list = [0 for num in range(256)]
for i in range(img.shape[0]):
    for j in range(img.shape[1]):
        pixel_list[img[i][j]] += 1

plt.bar(range(256), pixel_list)
plt.savefig("hist.png")
```

Result picture:



HW2-3

目標：

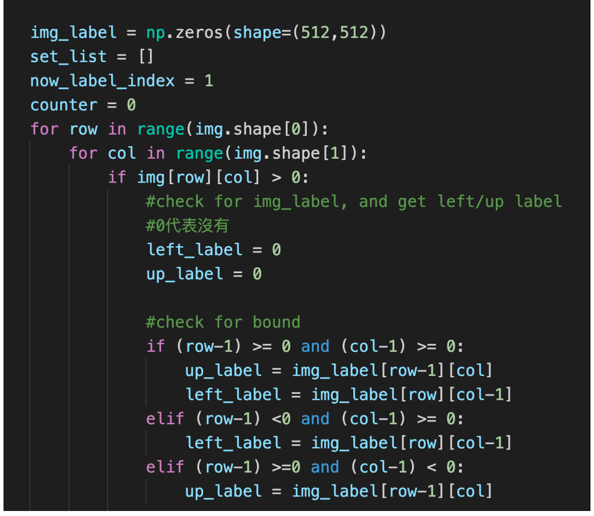
將照片的connected components找出來，並且用bounding box 與 centroid 表示在圖上

演算法：

我採用的是two-pass算法搭配4連通，首先先講解first pass步驟，再講解second pass的動作

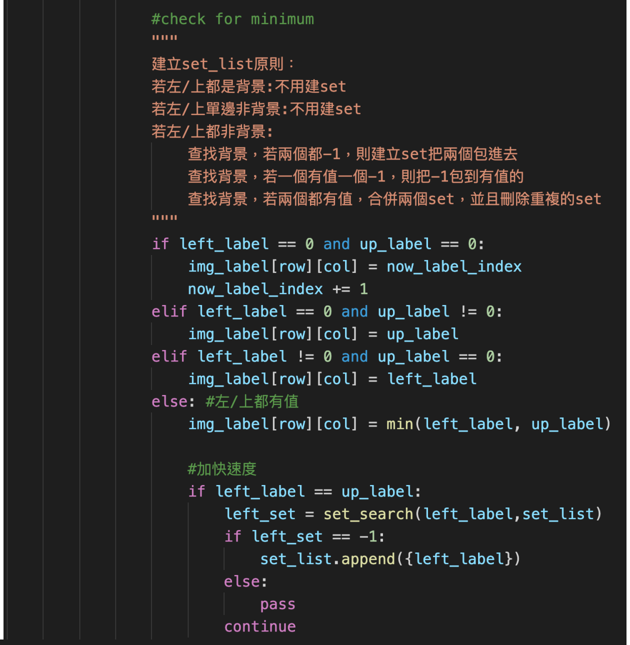
First pass:

我先拿binary過的照片（From HW-2-1），再來創建另外一個512*512的 zero matrix（為了做label），接下來掃過一次binary的照片，只要某個pixel有值，我就檢查這個pixel的”左邊”與”上面”pixel，如果都沒有label過的值，就賦予他一個新的值，若”左邊”或”上面”pixel已經有label，那就取兩者比較小的作為目前pixel的label，同時維護一個”哪個label與哪個label屬於同一個set”的資料，這個資料是為了two-pass可以從”同一個set”查找最小值



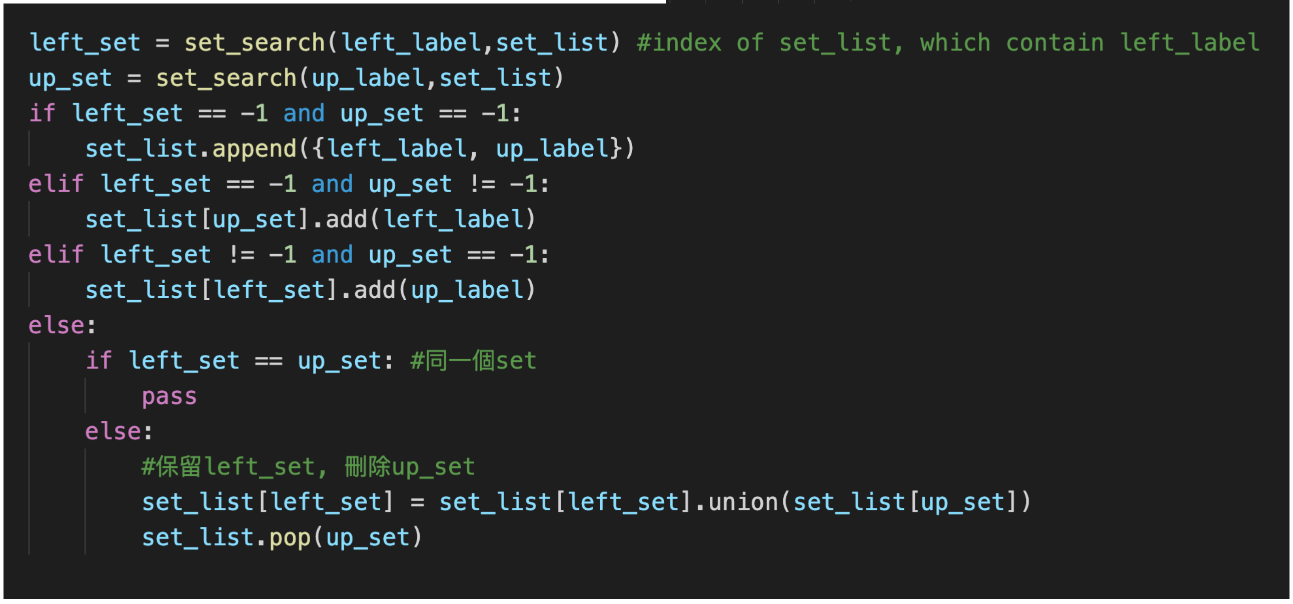
```python
img_label = np.zeros(shape=(512,512))
set_list = []
now_label_index = 1
counter = 0
for row in range(img.shape[0]):
    for col in range(img.shape[1]):
        if img[row][col] > 0:
            #check for img_label, and get left/up label
            #0代表沒有
            left_label = 0
            up_label = 0

            #check for bound
            if (row-1) >= 0 and (col-1) >= 0:
                up_label = img_label[row-1][col]
                left_label = img_label[row][col-1]
            elif (row-1) <0 and (col-1) >= 0:
                left_label = img_label[row][col-1]
            elif (row-1) >=0 and (col-1) < 0:
                up_label = img_label[row-1][col]
```



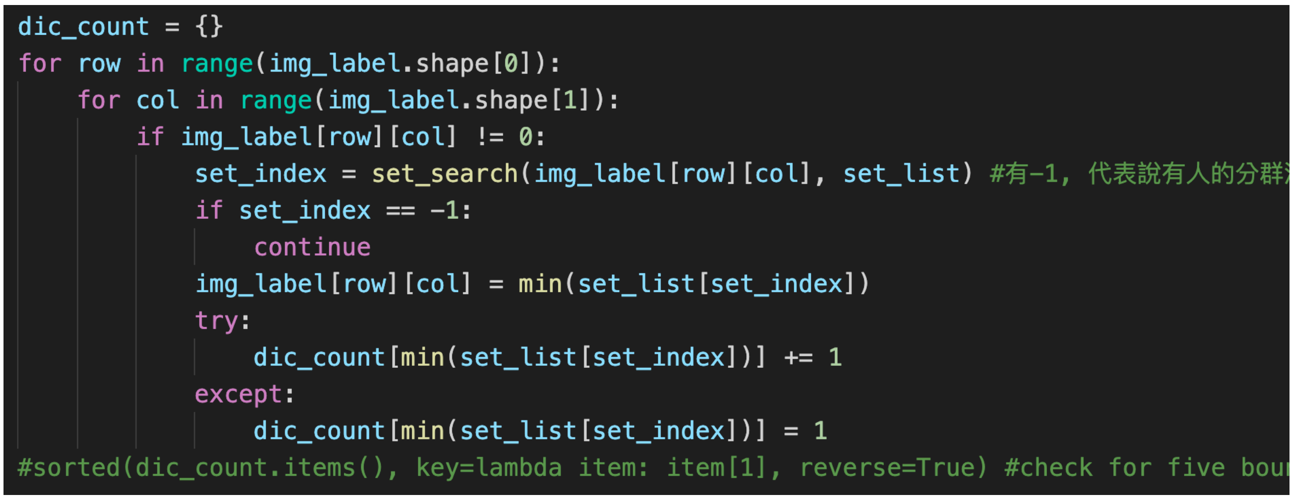
```python
                #check for minimum
                """
                建立set_list原則：
                若左/上都是背景:不用建set
                若左/上單邊非背景:不用建set
                若左/上都非背景:
                    查找背景，若兩個都-1，則建立set把兩個包進去
                    查找背景，若一個有值一個-1，則把-1包到有值的
                    查找背景，若兩個都有值，合併兩個set，並且刪除重複的set
                """
                if left_label == 0 and up_label == 0:
                    img_label[row][col] = now_label_index
                    now_label_index += 1
                elif left_label == 0 and up_label != 0:
                    img_label[row][col] = up_label
                elif left_label != 0 and up_label == 0:
                    img_label[row][col] = left_label
                else: #左/上都有值
                    img_label[row][col] = min(left_label, up_label)

                    #加快速度
                    if left_label == up_label:
                        left_set = set_search(left_label,set_list)
                        if left_set == -1:
                            set_list.append({left_label})
                        else:
                            pass
                        continue
```



```python
left_set = set_search(left_label,set_list) #index of set_list, which contain left_label
up_set = set_search(up_label,set_list)
if left_set == -1 and up_set == -1:
    set_list.append({left_label, up_label})
elif left_set == -1 and up_set != -1:
    set_list[up_set].add(left_label)
elif left_set != -1 and up_set == -1:
    set_list[left_set].add(up_label)
else:
    if left_set == up_set: #同一個set
        pass
    else:
        #保留left_set, 刪除up_set
        set_list[left_set] = set_list[left_set].union(set_list[up_set])
        set_list.pop(up_set)
```

Second pass:
重新掃過新的label matrix，然後透過查找之前維護的資料，選出這個label所屬的set之中的最小值，並且把label matrix當前的label換成這個最小值



```python
dic_count = {}
for row in range(img_label.shape[0]):
    for col in range(img_label.shape[1]):
        if img_label[row][col] != 0:
            set_index = set_search(img_label[row][col], set_list) #有-1, 代表說有人的分群
            if set_index == -1:
                continue
            img_label[row][col] = min(set_list[set_index])
            try:
                dic_count[min(set_list[set_index])] += 1
            except:
                dic_count[min(set_list[set_index])] = 1
#sorted(dic_count.items(), key=lambda item: item[1], reverse=True) #check for five bou
```

最後再找出bounding box, 也就是四個點，就可以完成這項作業了

```python
img_modified = cv2.imread('lena.bmp')
for i in range(len(boundbox_result)):

    boundbox_result[i][0] #min_x
    boundbox_result[i][1] #min_y
    boundbox_result[i][2] #max_x
    boundbox_result[i][3] #max_y
    leftup = (boundbox_result[i][0], boundbox_result[i][1])
    rightup = (boundbox_result[i][2], boundbox_result[i][1])
    leftdown = (boundbox_result[i][0], boundbox_result[i][3])
    rightdown = (boundbox_result[i][2], boundbox_result[i][3])

    leftcross = (boundbox_result[i][4]-10,boundbox_result[i][5])
    rightcross = (boundbox_result[i][4]+10,boundbox_result[i][5])
    upcross = (boundbox_result[i][4],boundbox_result[i][5]+10)
    downcross = (boundbox_result[i][4],boundbox_result[i][5]-10)

    cv2.line(img_modified, leftup, rightup, (0, 0, 255), 3)
    cv2.line(img_modified, rightup, rightdown, (0, 0, 255), 3)
    cv2.line(img_modified, rightdown, leftdown, (0, 0, 255), 3)
    cv2.line(img_modified, leftdown, leftup, (0, 0, 255), 3)
    cv2.line(img_modified, leftcross, rightcross, (0, 0, 255), 3)
    cv2.line(img_modified, upcross, downcross, (0, 0, 255), 3)
```

Result picture: