(a) a binary image(threshold at 128)

```
#讀取檔案 (numpy array)
img = cv2.imread('lena.bmp')
#Binariaze image at threshold
cv2.imwrite('binarize.bmp', img)#建立輸出的圖片檔

img1 = cv2.imread('binarize.bmp')
for i in range(512):
    for j in range(512):
        if int(img1[i][j][0]) >= 128:
            img1[i][j] = [255, 255, 255]
        else:
            img1[i][j] = [0, 0, 0]
cv2.imwrite('binarize.bmp', img1)
```

- 使用迴圈讀取lena.bmp的array，將vector從右下方開始讀取並寫入img1，最後存入binarize.bmp。

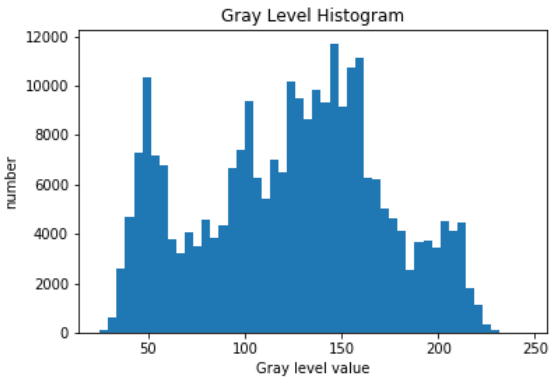
(b) a histogram

```
#Histogram
def draw_hist(myList,Title,Xlabel,Ylabel):
    plt.hist(myList, 50)
    plt.xlabel(Xlabel)
    plt.ylabel(Ylabel)
    plt.title(Title)
    plt.savefig("Histogram.png")
    plt.show()

data = [] #用list儲存數值
for i in range(512):
    for j in range(512):
```

```
        data.append(int(img[i][j][0]))
draw_hist(data,'Gray Level Histogram','Gray level value','number')
```

- 使用matplotlib.pyplot中的函式定義draw_hist的function，需傳入裝載資料的list,、histogram的標題、x座標名稱、y座標名稱。
- 讀取lena. bmp的array，將vector中的第一個數值存入data中。寫入完畢後，呼叫draw_hist將data中lena.bmp的gray level value以長條圖的方式呈現。



(c) connected component

```
#Connected components
labelArray = [[0 for _ in range(512)] for _ in range(512)]

#initialize each pixel to a new label
newlabel = 1
for i in range(512):
    for j in range(512):
        if int(img1[i][j][0]) != 0:
            labelArray[i][j] = newlabel
            newlabel += 1
```

- 使用課堂中介紹的第一個演算法：The iterative algorithm

  建立一個512*512的二維矩陣紀錄每一個pixel的label，先將矩陣都初始化為0。讀取img1中的array(binarize at 128後的結果)，若該vector的數值不為0，便將該vector對應的矩陣元素填上一個label，每填入一個label後都須將label的數值加一。

```
#Iteration of top-down followed by bottom-up
#使用四連通
while True:
    change = False
    #top-down pass
    for i in range(512):
        for j in range(512):
            if labelArray[i][j] != 0:
                u = 2700000
                l = 2700000
                if (i != 0) and (labelArray[i-1][j] != 0):#可以往上
```

```
                    u = labelArray[i-1][j]
                if (j != 0) and (labelArray[i][j-1] != 0):#可以往左
                    l = labelArray[i][j-1]
                m = min(u, l)
                if (m != 0) and (m < labelArray[i][j]):
                    change = True
                    labelArray[i][j] = m

    #bottom-up pass
    for i in range(511, -1, -1):
        for j in range(511, -1, -1):
            if labelArray[i][j] != 0:
                d = 2700000
                r = 2700000
                if (i != 511) and (labelArray[i+1][j] != 0):#可以往下
                    d = labelArray[i+1][j]
                if (j != 511) and (labelArray[i][j+1] != 0):#可以往右
                    r = labelArray[i][j+1]
                m = min(d, r)
                if(m != 0) and (m < labelArray[i][j]):
                    change = True
                    labelArray[i][j] = m
    if change == False:
        break
```

- 使用四連通的方式進行connected component計算

  Top-down的時候，檢查labelArray中的每一個元素上方、左方的鄰近元素是否為背景(label = 0)或者非0的label。若兩者皆為非0的label，則挑選出較小的label和該位置的label進行比較，若鄰近的label較小則將該位置更新為鄰近的label value；如果臨近的label皆為零則不需要改變；若有一邊label = 0，則只需與非0的label比較進行更新。

```
import itertools
from collections import Counter

# count the number of regions ...
test = list(itertools.chain(*labelArray))
sorted_dict =  Counter(test)
a = sorted(sorted_dict.items(), key=lambda x: x[1])
label_List = []
for i in range(len(a)-1, -1, -1):
    if(a[i][1] >= 500)and(a[i][0]!=0):
        label_List.append(a[i][0])
```

- 計算更新完的labelArray中各個label的總數。使用數值500進行過濾，將總數大於等於500個的label value加入label_List中。

```
#draw bounding box and centroid...
```

```
for l in range(len(label_List)):
    updown = []
    leftright = []
    for i in range(512):
        for j in range(512):
            if labelArray[i][j] == label_List[l]:
                updown.append(i)
                leftright.append(j)
    x = int(sum(leftright)/len(leftright))
    y = int(sum(updown)/len(updown))
    cv2.rectangle(img1, (min(leftright), min(updown)), (max(leftright),
max(updown)), (255, 0, 0), 1)
    cv2.circle(img1,(x, y), 4, (0, 0, 255), -1)
cv2.imwrite('connect_component.bmp', img1)
```

- 在labelArray計算label_List中各個label的row跟column，將該位置row的數值寫入updown的list中、colum的數值寫入leftright的list中。計算完同個label的所有row跟column後，由leftright和updown的最小值組成該label連通區域的右上頂點、leftright和updown的最大值組成左下頂點。將右上頂點、左下頂點、圖片檔案傳入cv2中的rectangle中畫出同個label的連通區域，反覆至畫完所有label_List中label的連通區域。計算連通區域的中心也是使用leftright和updown的list，將leftright和updown計算平均取整數，將數值和圖片檔案傳入cv2中的circle畫出該label連通區域的centroid。