
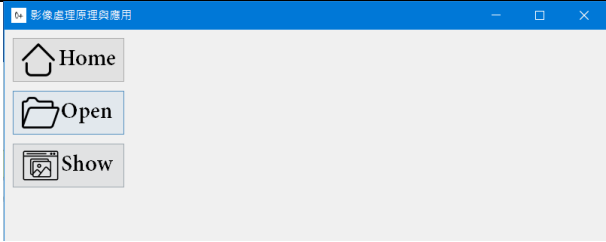
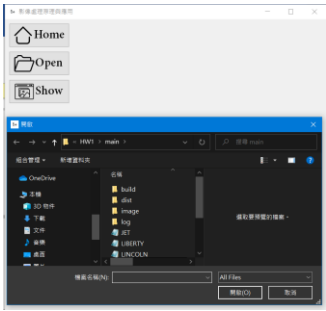
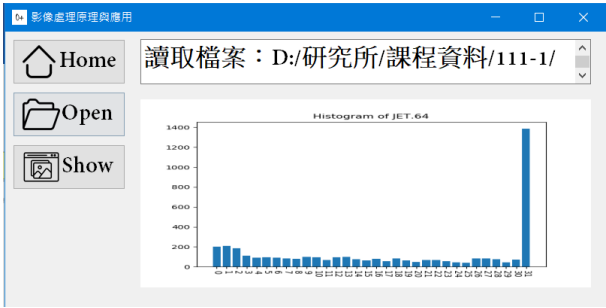



# 影像處理原理與應用作業

學號：R11631029 姓名：林正浩

## Part1:

UI 操作	
步驟	畫面
1 初始畫面 有 Read me 可以查看按鈕意思。	
2 點選 Part1，進入 Part1 的畫面	
3 點選 Open 讀取.64 檔案	
4 會顯示檔案位置跟 histogram	
5 點擊 Show 顯示圖片	

```

class Dot64_deal():
    # Read .64 file
    def Read64(self, filepath):
        self.filepath = filepath
        self.filename = self.filepath.split('/')[-1] # 取出檔案名稱
        with open(self.filepath) as photo64:
            self.content = photo64.read()
            # print(self.content)
            photo64.close()

    # 32 to 0-255
    dict32 = {'0' : 0, '1' : 1, '2' : 2, '3' : 3, '4' : 4, '5' : 5, '6' : 6, '7' : 7,
            '8' : 8, '9' : 9, 'A' : 10, 'B' : 11, 'C' : 12, 'D' : 13, 'E' : 14, 'F' : 15, 'G' : 16,
            'H' : 17, 'I' : 18, 'J' : 19, 'K' : 20, 'L' : 21, 'M' : 22, 'N' : 23, 'O' : 24, 'P' :
            25, 'Q' : 26, 'R' : 27, 'S' : 28, 'T' : 29, 'U' : 30, 'V' : 31}

    def photo64toarray(self):
        row = []
        self.content2array = []
        count = 0 # 用來計算是不是 64 列了
        for i in self.content:
            if(count == 64):
                break
            if(i != '\n'):
                try:
                    row.append((self.dict32[i] + 1 ) * 8 - 1)
                except:
                    row.append(i)
            else:
                self.content2array.append(row)
                count += 1
                row = []
            if(i == '\x1a'): # deal some text which include '\x1a' in the ead.
                row.remove('\x1a')
                self.content2array.append(row)
                count += 1
                break
        self.content2array = np.array(self.content2array)

    # Statistics the number of every symbols
    def statistics(self):
        self.Statistics = []

```

```

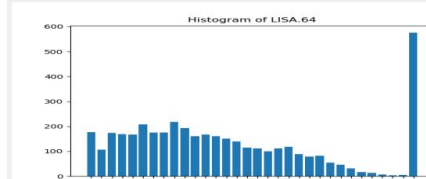
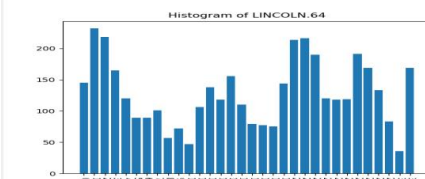
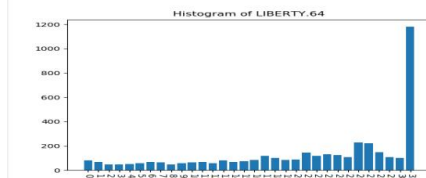
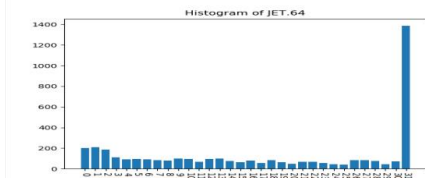
for i in self.dict32.keys():
    Count = 0
    for j in self.content:
        if(i == j):
            Count += 1
    self.Statistics.append(Count)
self.Statistics = np.array(self.Statistics)

# Drawing the hist gram
def histogram(self, Static):
    num = np.array(tuple(range(32)))
    # num = (num + 1 ) * 8 - 1
    plt.bar(range(32), Static)
    plt.title('Histogram of ' + self.filename)
    plt.xticks(range(32), num, rotation = -90)
    plt.savefig('log/part1_histogram.png')
    plt.close()


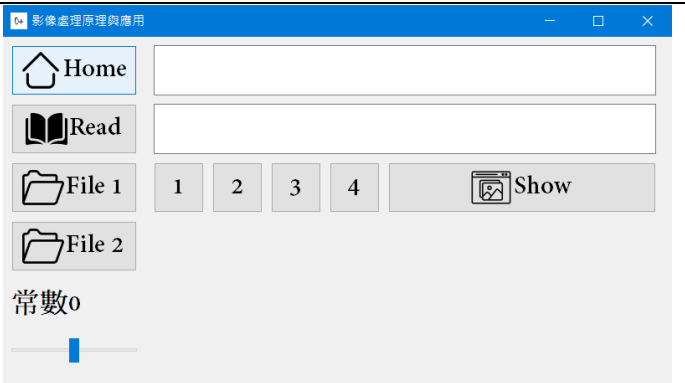
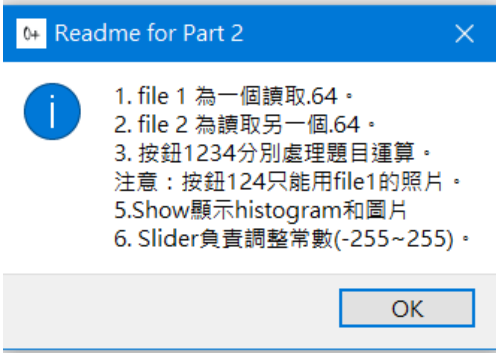
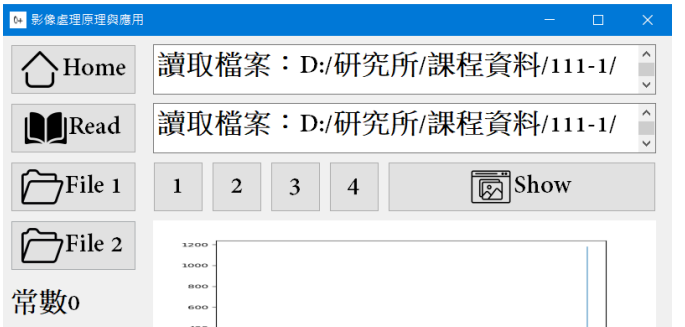
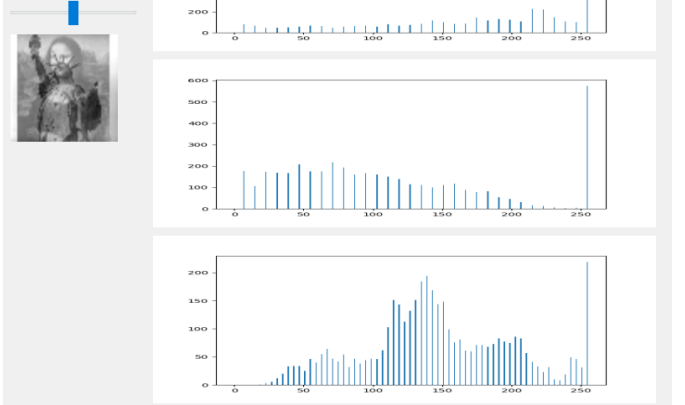
```

1. 方法 Read64()，進行.64 檔案的讀取，將讀取內容存成 string 的 type
2. 方法 photo64toarray()，將讀取的 string 轉換成 2D array，每一個元素轉換成 0-255 的 int
3. 方法 statistics()，進行內容的統計
4. 方法 histogram()，將 3 的統計畫成 histogram 並存檔下來以便顯示在 UI 上面。

## 結果



## Part2:

UI 操作	
步驟	畫面
<b>1 初始畫面</b> 有 Read me 可以查看按鈕意思。	
<b>2 點選 Part2，進入 Part2 的畫面</b>	
<b>3 不知道操作先點擊 Read 查看按鈕意義</b>	
<b>4 按照 Read 指事，要運行：</b> Op1:加減常數 Op2:常數相乘 Op4:列相減 請使用 File1 的檔案 Op3:照片合併 請使用 File1，File2 的檔案	
<b>5 點擊 Show 顯示質方圖與圖片</b> 圖片會顯示在左方 Histogram 依序顯示為： File1 histogram File2 histogram(Op3 才會顯示) Histogram after operation	

## 重要演算法(擷取部分程式碼)

```
# Part 2 運算用的 4 個 method

## 運算 1：影片加減一個常數-->調整明暗
def op1(self, Array1, constant):
    h, w = Array1.shape
    for i in range(h):
        for j in range(w):
            value = Array1[i][j] + constant
            if(value > 255):
                Array1[i][j] = 255
            elif(value < 0):
                Array1[i][j] = 0
            else:
                Array1[i][j] = round(value)
    return Array1

## 運算 2：乘上一個常數
def op2(self, Array2, constant):
    h, w = Array2.shape
    for i in range(h):
        for j in range(w):
            value = Array2[i][j] * constant
            if(value > 255):
                Array2[i][j] = 255
            elif(value < 0):
                Array2[i][j] = 0
            else:
                Array2[i][j] = round(value)
    return Array2

## 運算 3：兩矩陣平均
def op3(self, Array1, Array2):
    Array = np.round((Array1 + Array2) / 2.)
    return Array

## 運算 4：f(x,y) - f(x-1,y)
def op4(self, Array4):
    h, w = Array4.shape
    for i in range(1, h):
        for j in range(w):
            value = Array4[i][j] - Array4[i - 1][j]
```

```

if(value > 255):
    Array4[i][j] = 255
elif(value < 0):
    Array4[i][j] = 0
else:
    Array4[i][j] = round(value)
return Array4

```

Op1: 透過 numpy lib 實現 array 每一元素加上 Slider 調整的常數

Op2: 同上，每一元素乘上 Slider 調整的常數

Op3: 透過 numpy lib 實現 array 的相加取平均

Op4: 利用兩個 for loop 取樣並減掉 x-1 的元素值

\* 以上運算有透過 if else 判斷：

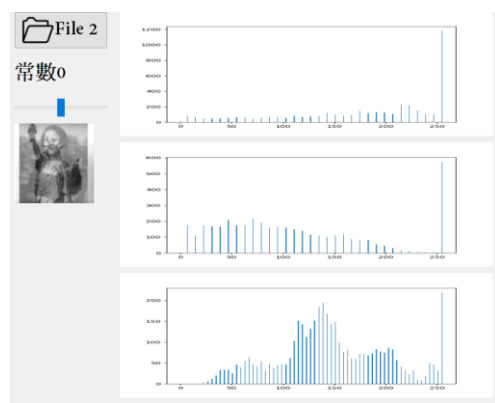
超過 255→定義 255

小於 0→定義 0

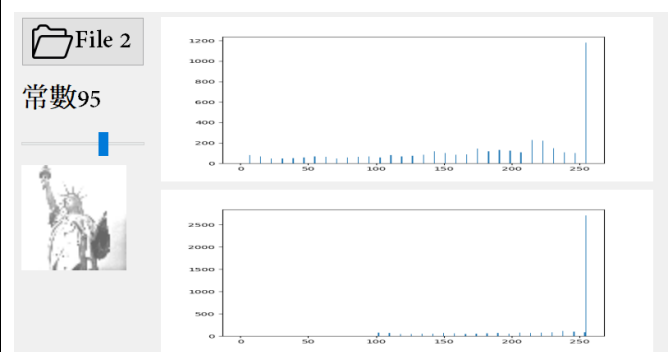
都沒有就是計算後的數值

## 結果

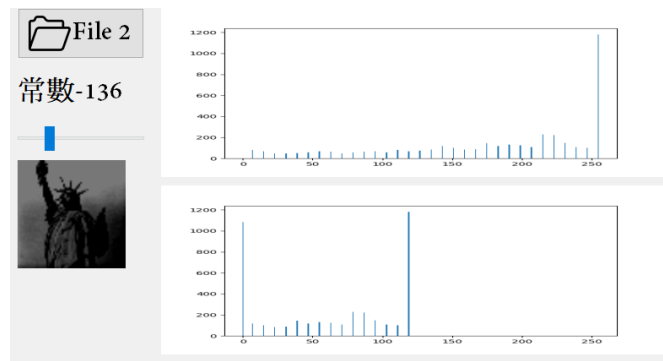
### OP3



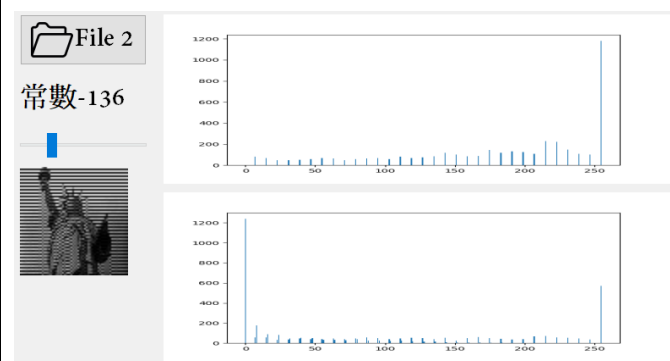
### OP1

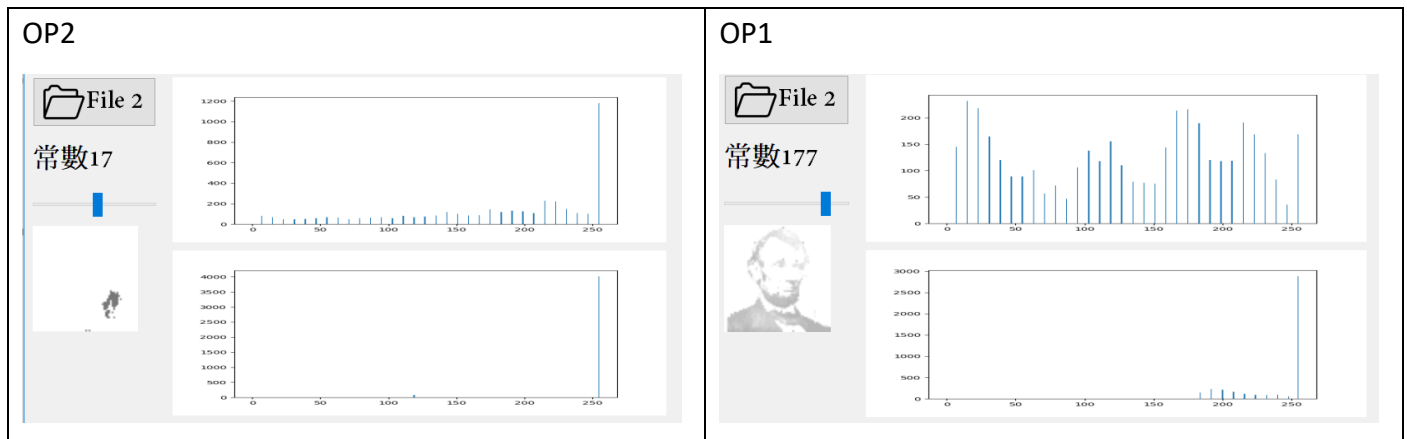


### OP1



### OP4





## 討論：

Part1:

透過 part1 的練習可以了解到 histogram 的演算方法。

Part2:

Op1: 可以進行亮暗的調整，histogram 會進行平移。

Op2: 根據目前定義的 Slider 範圍(-255~255, integer)可能沒有辦法看出作用。

Op3: 可以看到兩張圖片同時顯示。

Op4: 可以看到 0 的 histogram 數量增加，畢竟一張圖片當中元素附近的數值都非常相似，故圖片會呈現柵欄的樣子。