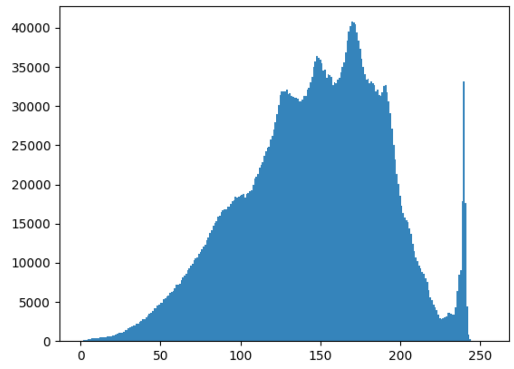
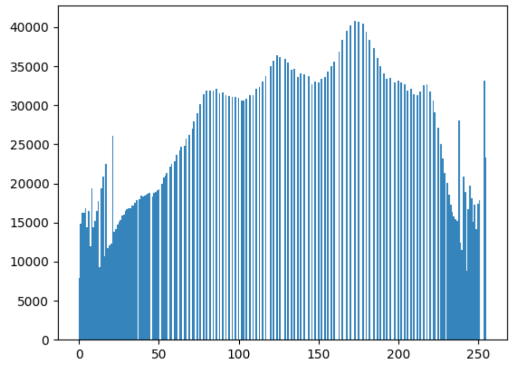
```
import os
import cv2 as cv2
import numpy as np
from matplotlib import pyplot as plt

def histogram(img):
    histogram_ = np.zeros(256, dtype=np.uint32) # 存放讀到的 pixel 值
    for k in range(len(histogram_)):
        histogram_[k] = np.sum(img.flat == k)     # 將符合 SUM 起來
    return histogram_

def equalize(img,histogram_):
    def sumcdf(histogram_):
        cdf = np.zeros_like(histogram_, dtype=np.uint32) # 累積分布函數（cdf）
        cdf[0] = histogram_[0] # 因為下方 cdf[k-1], 在 cdf[0] 時前面沒有值,有
可能會讀到亂七八糟的東西,因此要先將[0]先寫進去
        for k in range(len(cdf)):
            cdf[k] = cdf[k-1] + np.sum(img.flat == k) # 取的 cdf 累計"值"
        return cdf

    cdf_pro = sumcdf(histogram_) / (img.shape[0] * img.shape[1]) # 算出 cdf 的機
率(probability)
    cdf_value = np.zeros_like(img, dtype=np.uint8) # 存放乘以 CDF 的 pixel 值
    for row in range(img.shape[0]):
        for col in range(img.shape[1]):
            cdf_value[row,col] = np.uint8(np.round((cdf_pro[img[row,col]] -
np.min(cdf_pro)) / (np.max(cdf_pro) - np.min(cdf_pro)) * 255))
           # 將 row col 的值,乘上對應 cdf 出現的機率值 = 均化
    return cdf_value   # 將均化後的值回傳

def gray():
    img_path='E:/Program_File/PYTHON/數位影像處理作業/HW_1/snow.jpg'
    img_filepath = os.path.splitext(img_path)[0]      # 拆分路徑 & 副檔名，0 為
路徑
    img_fileextension = os.path.splitext(img_path)[1]   # 1 為副檔名
    img_filename = os.path.basename(img_filepath)       # 取出檔名不含副檔名

    img = cv2.imread(img_filename+img_fileextension,cv2.IMREAD_GRAYSCALE)
    histogram_ = histogram(img)
    cdf_value = equalize(img,histogram_)
    histogram_equ = histogram(cdf_value)
    plt.bar(range(len(histogram_)), histogram_, alpha=0.9, width = 1, lw=1)
    plt.figure()
    plt.bar(range(len(histogram_)), histogram_equ, alpha=0.9, width = 1, lw=1)
    plt.show()
    plt.close()
    cv2.imshow('img',img)
    cv2.imshow('equalize',cdf_value)
    cv2.waitKey()
    cv2.destroyAllWindows()
    cv2.imwrite(img_filename+'_gray'+img_fileextension,img)
    cv2.imwrite(img_filename+'_gray_equalize'+img_fileextension,cdf_value)

def multicolor():
    img_path='E:/Program_File/PYTHON/數位影像處理作業/HW_1/star.jpg'
    img_filepath = os.path.splitext(img_path)[0]      # 拆分路徑 & 副檔名，0 為路徑
    img_fileextension = os.path.splitext(img_path)[1]   # 1 為副檔名
    img_filename = os.path.basename(img_filepath)       # 取出檔名不含副檔名

    img = cv2.imread(img_filename+img_fileextension,-1)   # 讀檔
    img_b, img_g, img_r = cv2.split(img)                  # 將對檔案分割，順序為BGR，而非 RGB

    img_b_hist = histogram(img_b)                 # 分別將 BGR 做 histogram
    img_g_hist = histogram(img_g)
    img_r_hist = histogram(img_r)

    img_b_eq = equalize(img_b, img_b_hist)       # 各別接著做 equalize
    img_g_eq = equalize(img_g, img_g_hist)
    img_r_eq = equalize(img_r, img_r_hist)

    img_b_eq_hist = histogram(img_b_eq)       # 最後將 equalize 後的 BGR 做histogram
    img_g_eq_hist = histogram(img_g_eq)
    img_r_eq_hist = histogram(img_r_eq)
    plt.subplot(3,2,1)                              #  子圖，( 3 * 2 (列 column *  行 row),  第一個圖，左上)
    plt.bar(np.arange(img_b_hist.shape[0]), img_b_hist, alpha=0.9, width = 1, lw=1, align = 'center',color = 'b')
    plt.subplot(3,2,2)                              #  子圖，( 3 * 2 ,  第二個圖，右上)
    plt.bar(np.arange(img_b_eq_hist.shape[0]), img_b_eq_hist, alpha=0.9, width = 1, lw=1, align = 'center',color = 'b')
    plt.subplot(3,2,3)                              #  子圖，( 3 * 2 ,  第三個圖，左中)
    plt.bar(range(len(img_g_hist)), img_g_hist, alpha=0.9, width = 1, lw=1, align = 'center',color = 'g')
    plt.subplot(3,2,4)
    plt.bar(range(len(img_g_eq_hist)), img_g_eq_hist, alpha=0.9, width = 1, lw=1, align = 'center',color = 'g')
    plt.subplot(3,2,5)
    plt.bar(range(len(img_r_hist)), img_r_hist, alpha=0.9, width = 1, lw=1, align = 'center',color = 'r')
    plt.subplot(3,2,6)
    plt.bar(range(len(img_r_eq_hist)), img_r_eq_hist, alpha=0.9, width = 1, lw=1, align = 'center',color = 'r')

    plt.show()
    plt.close()        #  開啟圖片後要記得關閉，當有大量的圖片或資料開啟後，記憶體可能會爆，因此要手動關閉他

    img_merge = cv2.merge([img_b_eq,img_g_eq,img_r_eq])        #  將均化後的 BGR  合併
    cv2.imwrite(img_filename+'_merge_equalize'+img_fileextension,img_merge)

    cv2.imshow('img',img)
    cv2.imshow('equalize',img_merge)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

# gray()
# multicolor()
```

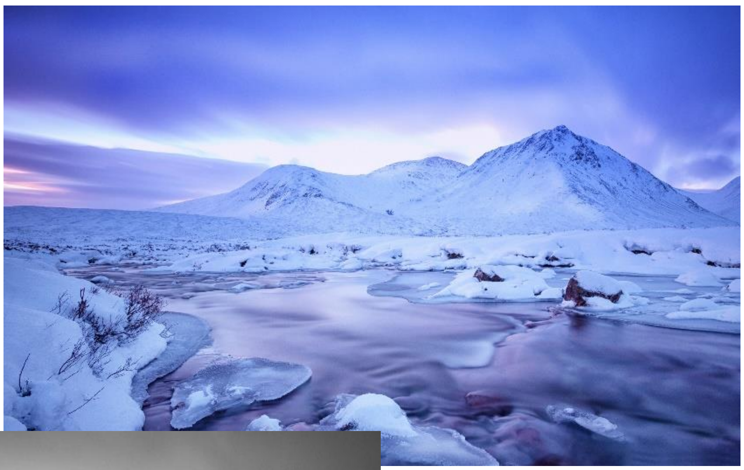
GRAY :

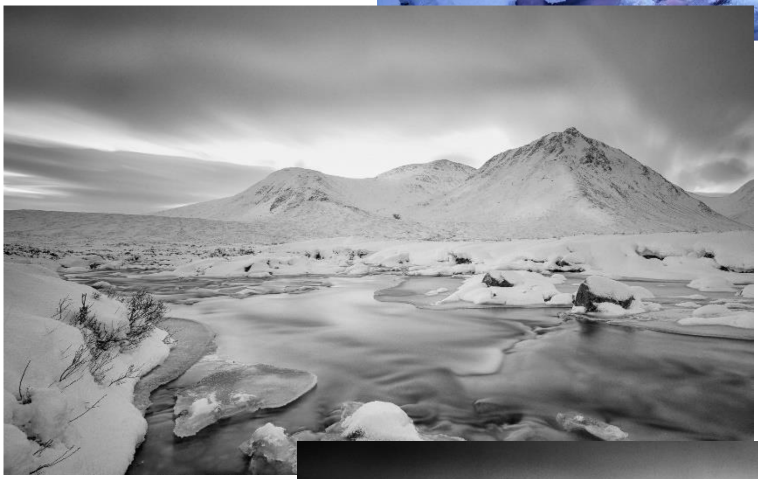


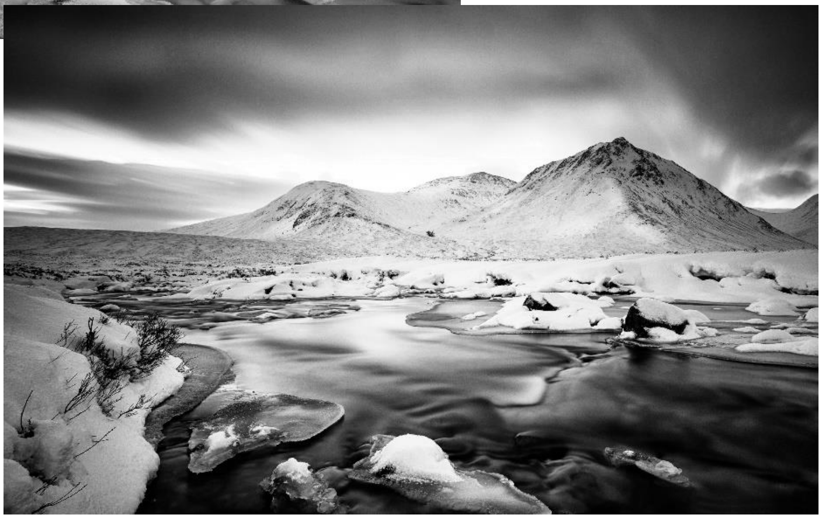
40000
35000
30000
25000
20000
15000
10000
5000
0
0
50
100
150
200
250



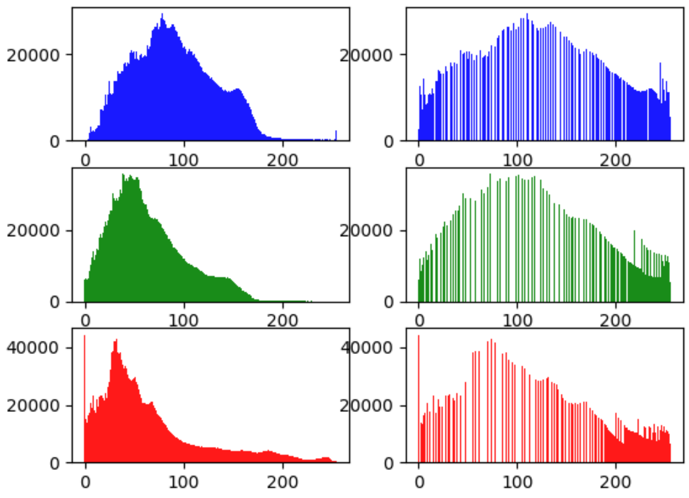
40000
35000
30000
25000
20000
15000
10000
5000
0
0
50
100
150
200
250

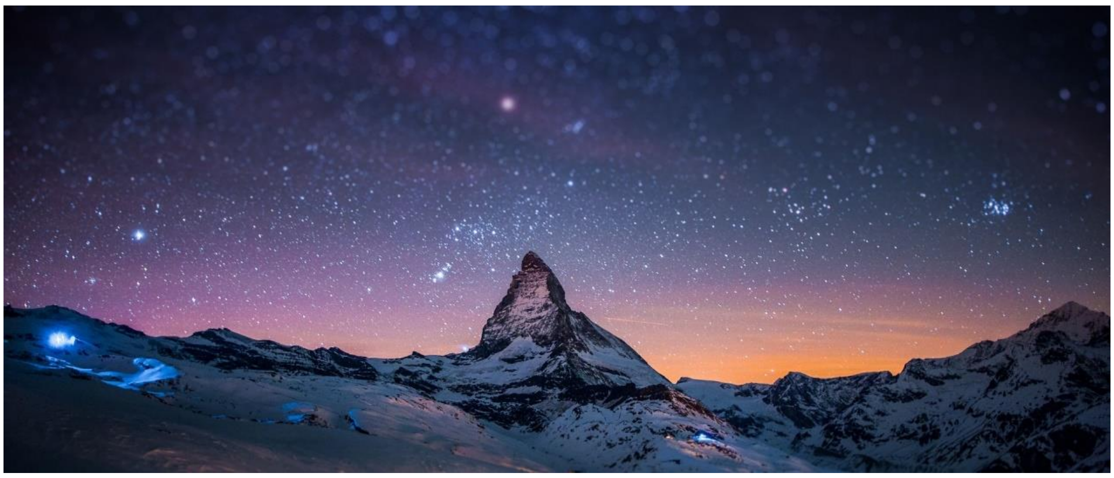


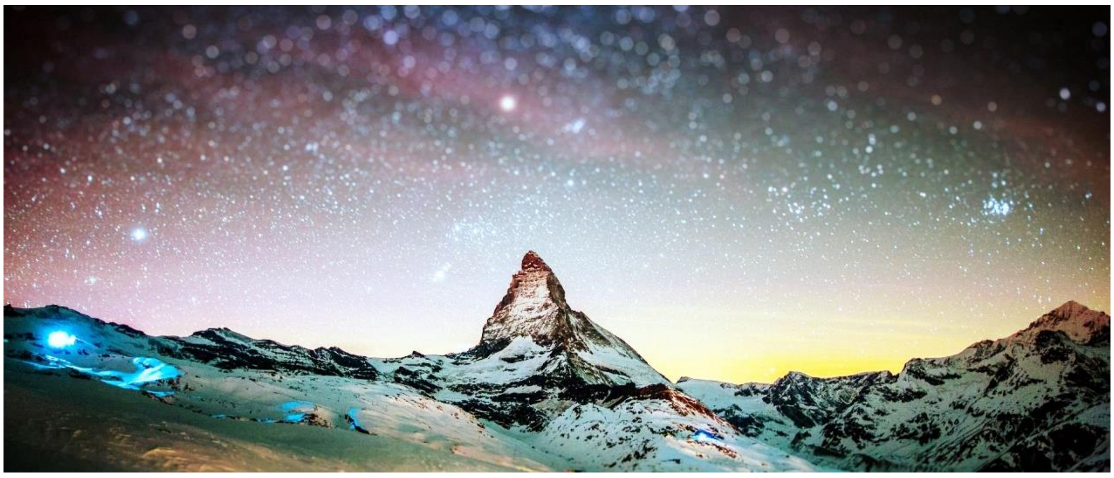




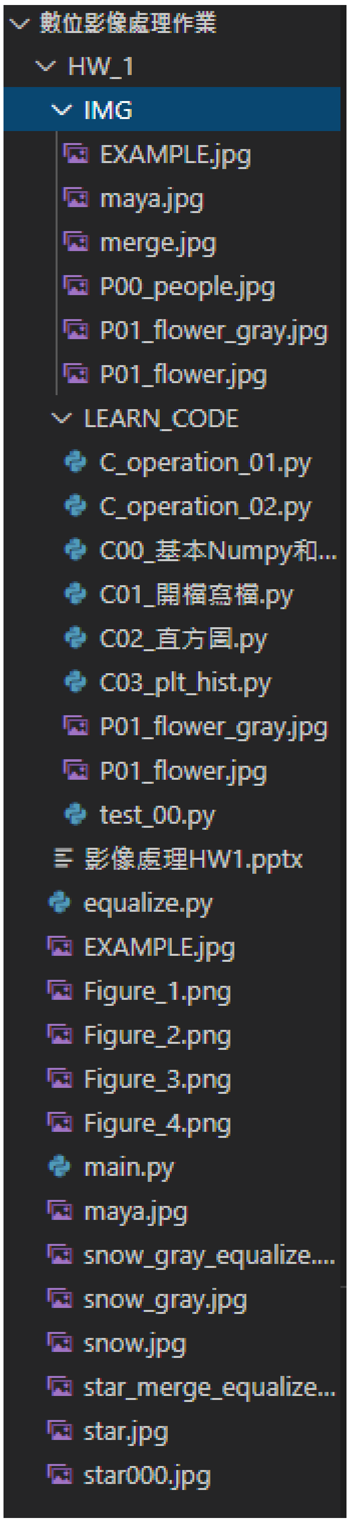
multicolor :


20000
0
0
100
200
20000
0
0
100
200
20000
0
0
100
200
20000
0
0
100
200
40000
20000
0
0
100
200
40000
20000
0
0
100
200





心得 ：
在做直方圖以及均化之前對 numpy、cv2、matplotlib，完全沒有概念，花了許多時間對其各種用法進行嘗試。
在寫直方圖時整體還算順利，但是當要做均化時我參考了維基百科上的數學公式，有做出 CDF，在我以為我已經成功時，跑出來的圖卻並不正確，因為實際上我只有算出機率，並沒有乘上原本的值，在與同學請教過後，才成功的求出均化後的值並成功輸出



```
數位影像處理作業
  HW_1
    IMG
      EXAMPLE.jpg
      maya.jpg
      merge.jpg
      P00_people.jpg
      P01_flower_gray.jpg
      P01_flower.jpg
    LEARN_CODE
      C_operation_01.py
      C_operation_02.py
      C00_基本Numpy和...
      C01_開檔寫檔.py
      C02_直方圖.py
      C03_plt_hist.py
      P01_flower_gray.jpg
      P01_flower.jpg
      test_00.py
    影像處理HW1.pptx
    equalize.py
    EXAMPLE.jpg
    Figure_1.png
    Figure_2.png
    Figure_3.png
    Figure_4.png
    main.py
    maya.jpg
    snow_gray_equalize....
    snow_gray.jpg
    snow.jpg
    star_merge_equalize...
    star.jpg
    star000.jpg
```