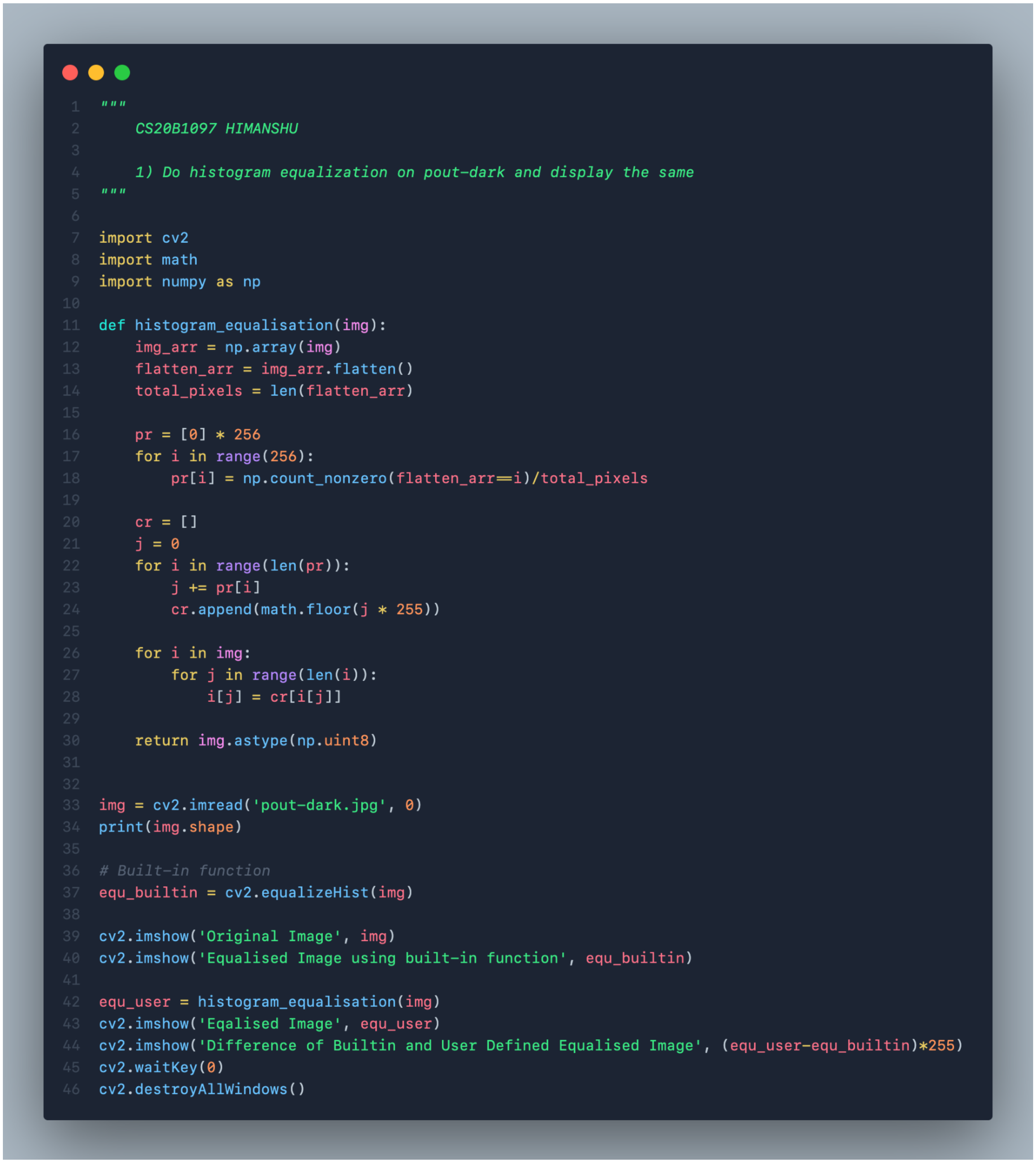
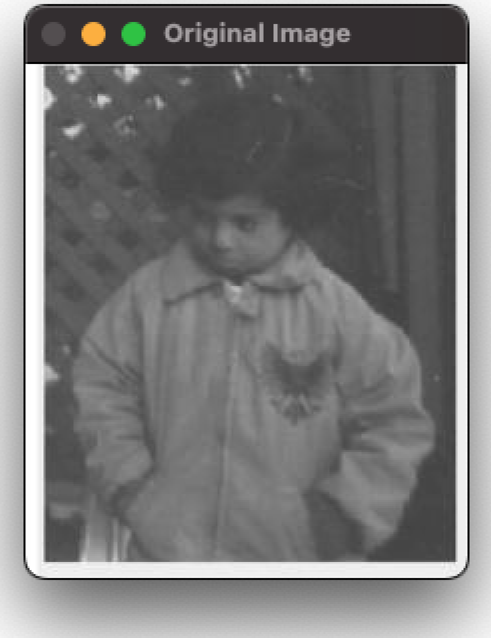
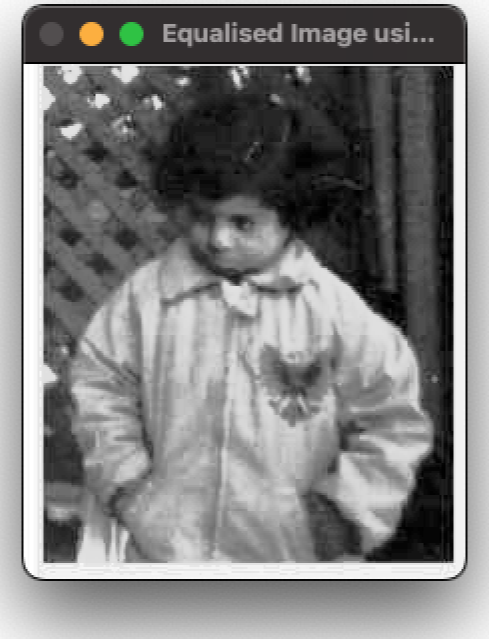
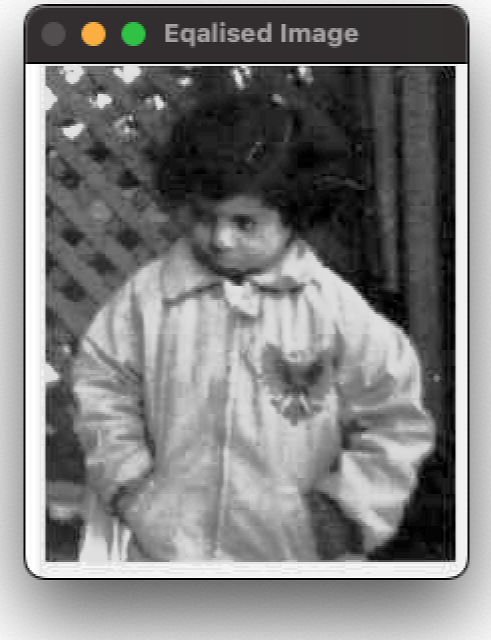
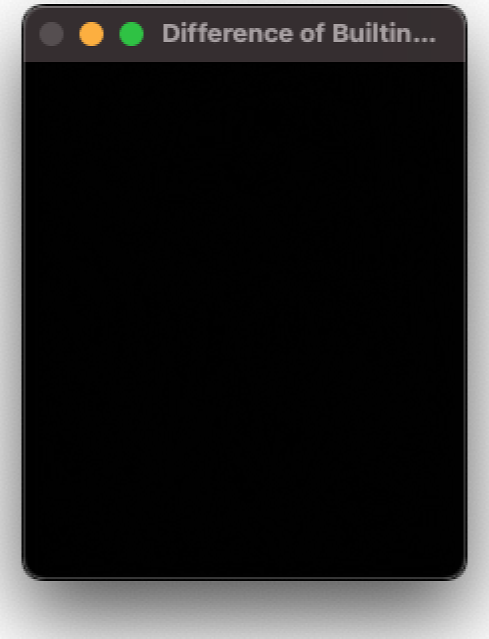


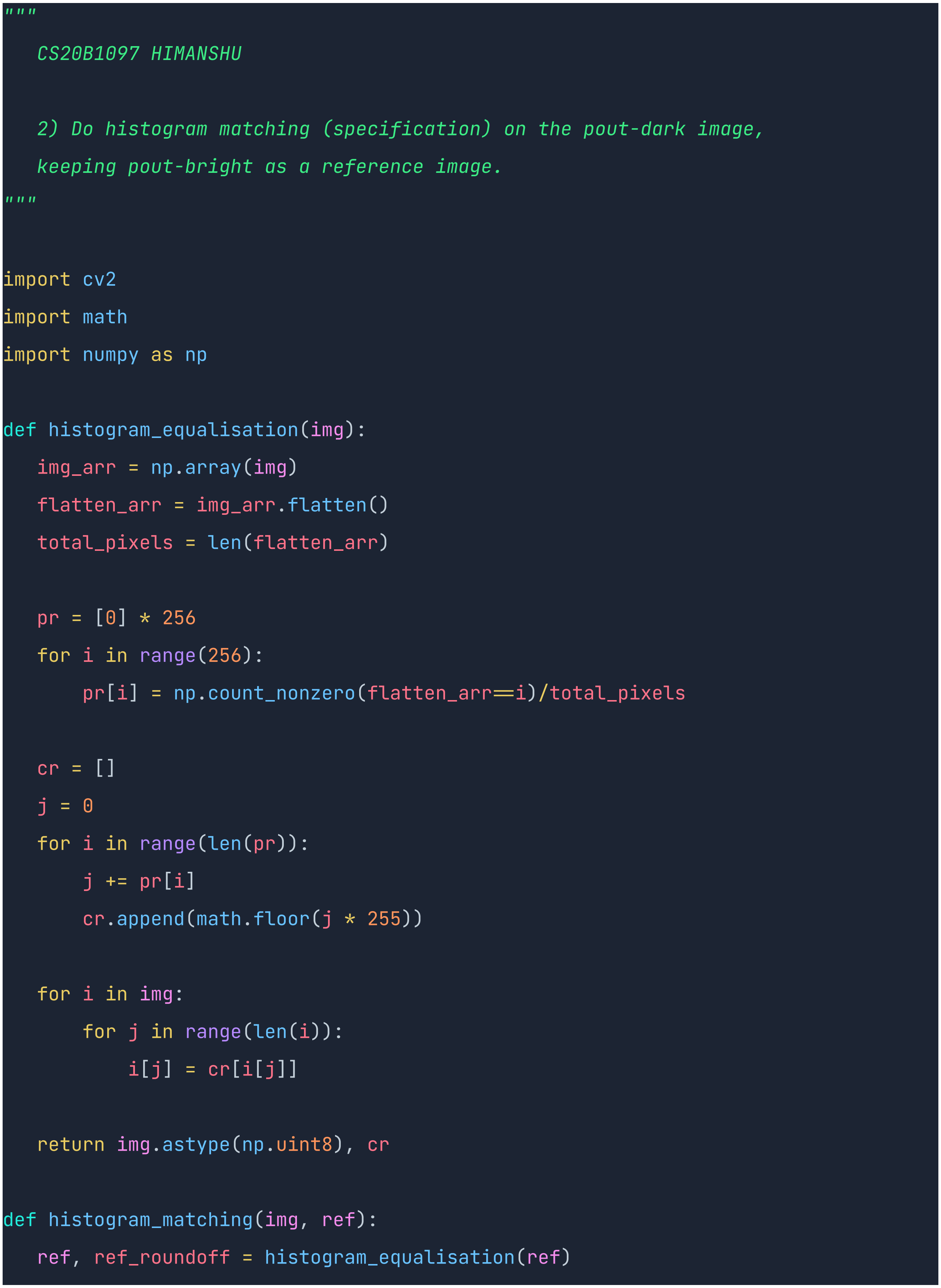
```python
"""
    CS20B1097 HIMANSHU

    1) Do histogram equalization on pout-dark and display the same
"""

import cv2
import math
import numpy as np

def histogram_equalisation(img):
    img_arr = np.array(img)
    flatten_arr = img_arr.flatten()
    total_pixels = len(flatten_arr)

    pr = [0] * 256
    for i in range(256):
        pr[i] = np.count_nonzero(flatten_arr==i)/total_pixels

    cr = []
    j = 0
    for i in range(len(pr)):
        j += pr[i]
        cr.append(math.floor(j * 255))

    for i in img:
        for j in range(len(i)):
            i[j] = cr[i[j]]

    return img.astype(np.uint8)


img = cv2.imread('pout-dark.jpg', 0)
print(img.shape)

# Built-in function
equ_builtin = cv2.equalizeHist(img)

cv2.imshow('Original Image', img)
cv2.imshow('Equalised Image using built-in function', equ_builtin)

equ_user = histogram_equalisation(img)
cv2.imshow('Eqalised Image', equ_user)
cv2.imshow('Difference of Builtin and User Defined Equalised Image', (equ_user-equ_builtin)*255)
cv2.waitKey(0)
cv2.destroyAllWindows()
```


Original Image

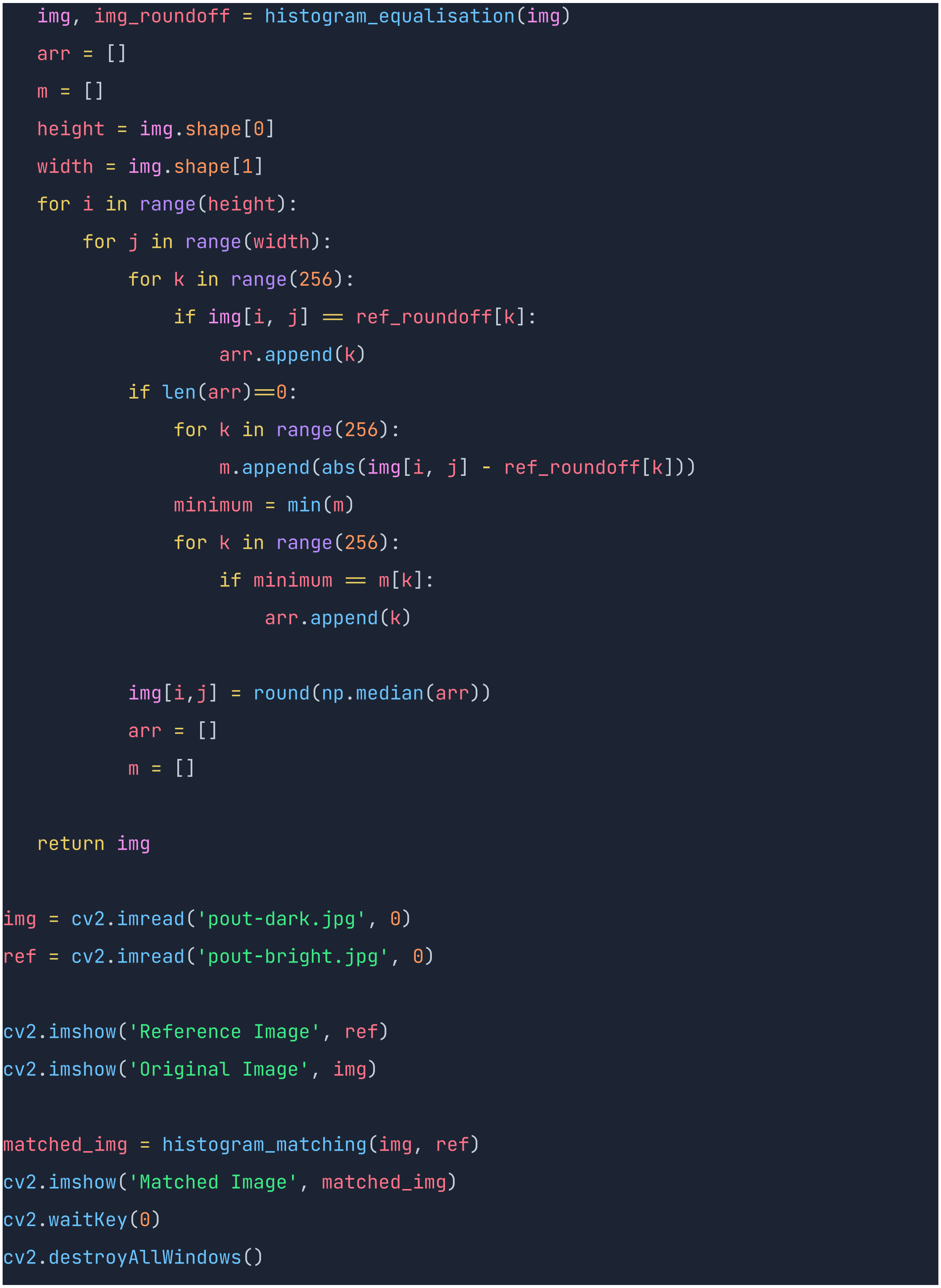

Equalised Image usi...


Eqalised Image

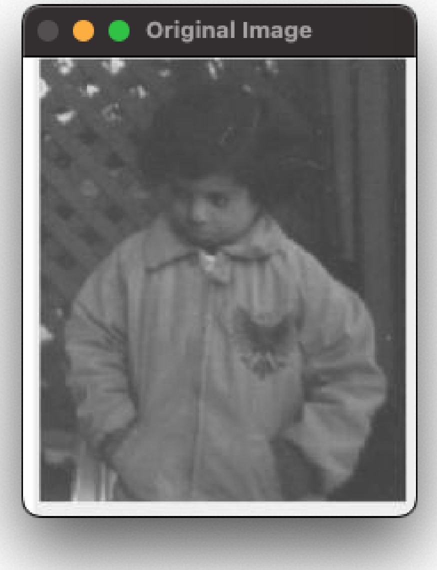

Difference of Builtin...

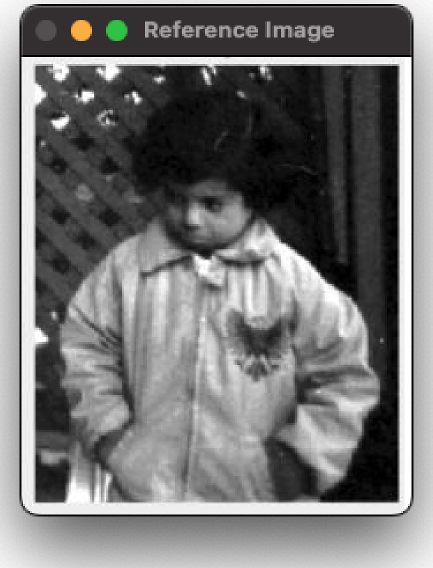


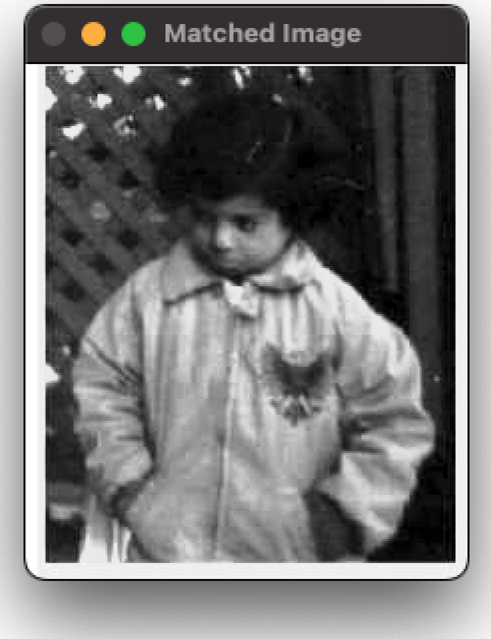
```
"""
    CS20B1097 HIMANSHU

    2) Do histogram matching (specification) on the pout-dark image,
    keeping pout-bright as a reference image.
"""

import cv2
import math
import numpy as np

def histogram_equalisation(img):
    img_arr = np.array(img)
    flatten_arr = img_arr.flatten()
    total_pixels = len(flatten_arr)

    pr = [0] * 256
    for i in range(256):
        pr[i] = np.count_nonzero(flatten_arr==i)/total_pixels

    cr = []
    j = 0
    for i in range(len(pr)):
        j += pr[i]
        cr.append(math.floor(j * 255))

    for i in img:
        for j in range(len(i)):
            i[j] = cr[i[j]]

    return img.astype(np.uint8), cr

def histogram_matching(img, ref):
    ref, ref_roundoff = histogram_equalisation(ref)
```



```
    img, img_roundoff = histogram_equalisation(img)
    arr = []
    m = []
    height = img.shape[0]
    width = img.shape[1]
    for i in range(height):
        for j in range(width):
            for k in range(256):
                if img[i, j] == ref_roundoff[k]:
                    arr.append(k)
            if len(arr)==0:
                for k in range(256):
                    m.append(abs(img[i, j] - ref_roundoff[k]))
                minimum = min(m)
                for k in range(256):
                    if minimum == m[k]:
                        arr.append(k)

            img[i,j] = round(np.median(arr))
            arr = []
            m = []

    return img

img = cv2.imread('pout-dark.jpg', 0)
ref = cv2.imread('pout-bright.jpg', 0)

cv2.imshow('Reference Image', ref)
cv2.imshow('Original Image', img)

matched_img = histogram_matching(img, ref)
cv2.imshow('Matched Image', matched_img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```


Original Image


Reference Image


Matched Image