문제3

2018316501 박민지

3. Eh5.HistogramSpecification.py 소스의 맨 마지막 줄 미션
미션: 새 창을 열고 src/template/output 영상의 RGB별 히스토그램을 그려 보자.
조건: rgb 히스토그램 데이터를 반환하는 다음 함수를 제작한 후 코딩할 것
r, g, b = myHist(img)
img는 3채널 영상. 부동소수와 uint8을 구분하지 않고 모두 지원할 것
r, g, b는 각각 최댓값에 의해 정규화된 0~1의 분포도.

```
s_file = 'monarch.png'; t_file = 'forest-park-trail.jpg'
```

---원본영상을 monarch.png, template영상을 forest-park-trail.jpg로 설정 합니다.

```
def MyHist(image, reference, multichannel=False):
    if multichannel:      # multichannel이 무언가가 들어가 있을 때, True 일때
        matched = np.empty(image.shape, dtype=image.dtype)
        for channel in range(image.shape[-1]):
            matched_channel = _match_cumulative_cdf(image[..., channel],
                                                    reference[..., channel])
            matched[..., channel] = matched_channel
    else:
        matched = _match_cumulative_cdf(image, reference)

    return matched
```

--- MyHist 함수를 생성합니다. Multichannel이 true일 때 matched에 image.shape를 넣어줍니다.

Channel에 image.shape의 -1까지 루프를 돌려 matched_channel에 skimage에서 제공하는 _match_cumulative_cdf함수를 넣어줍니다.

그리고 matched를 반환해 줍니다.

```
def ncdf(im):
    cdf, b = cumulative_distribution(im)
```

```
    for i in range(b[0]):
        cdf = np.insert(cdf, 0, 0)
    for i in range(b[-1]+1, 256):
        cdf = np.append(cdf, 1)
    return cdf
```

--- ncdf 함수를 생성해 입력받은 영상의 ncdf를 반환해 줍니다.

```
im = imread(path + s_file)        # 원본 영상
im_t = imread(path + t_file)      # template 영상
im1 = MyHist(im, im_t, multichannel=True)
```

--- 원본영상을 im에 저장하고 template영상을 im_t에 저장합니다. Im1에는 위에서 만들었던 MyHist 함수를 사용해 원본영상과 template영상의 output된 영상을 저장합니다.

```
plt.subplot(221)
plt.imshow(im1[...,:3])
plt.axis('off')
plt.title('Output Image')
```

--- Output Image로 타이틀을 입력 후 격자를 없앤 다음에 3채널로 이미지를 보여줍니다.

```
clr_title = ['Red', 'Green', 'Blue']

for i in range(3):              # RGB 채널에 대해 loop 작업
    c = ncdf(im[..., i])        # 원본의 cdf
    c_t = ncdf(im_t[..., i]) # template의 cdf
    c_r = ncdf(im1[...,i])    # 결과의 cdf

    plt.subplot(222+i)

    plt.plot(c * 255, 'r', label='original(ncdf)')
    plt.plot(c_t * 255, 'g', label='template(ncdf)')
    plt.plot(c_r * 255, '--b', label='result(ncdf)')
```

```
        plt.title(clr_title[i])
        plt.grid('on'), plt.legend()
```

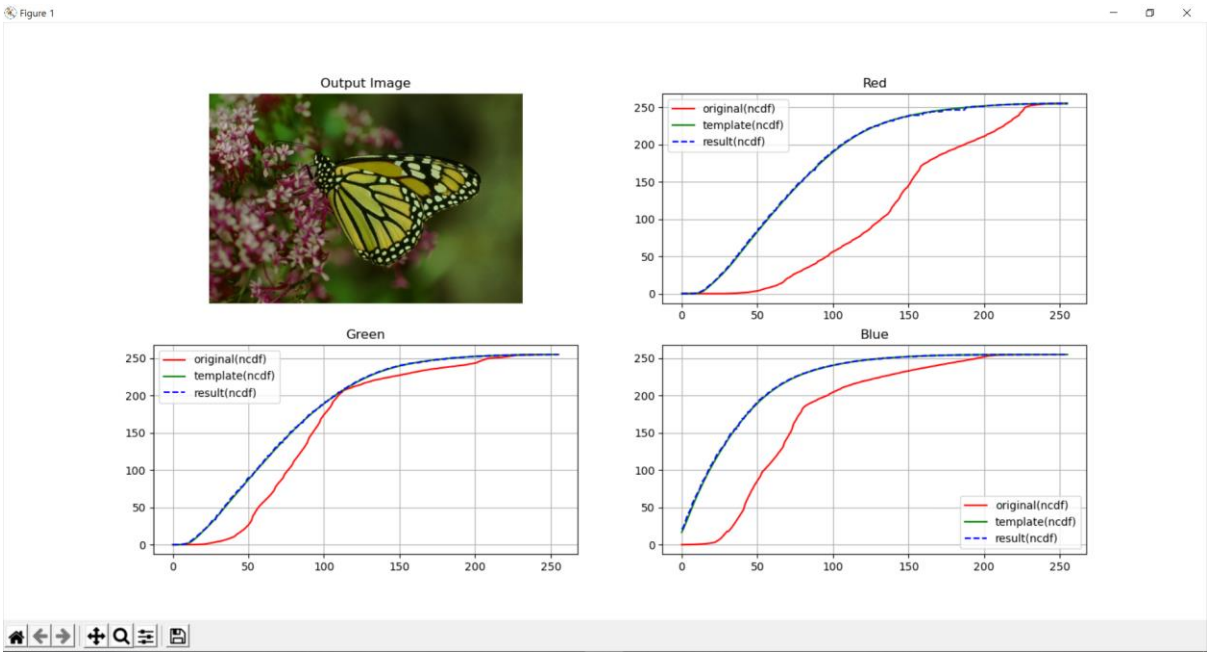
--- R,G,B에 관하여 for문으로 loop 작업을 해줍니다.

```
plt.savefig('./result.png')
```

--- plt로 출력될 영상을 png로 저장을 합니다.

```
plt.show()
exit(0)
```

--- 그리고 영상을 출력합니다.



--- 빨간색선으로 원본영상의 RGB를 표현하고, 초록색 선으로 template의 RGB를 표현했습니다. 마지막으로 파란색 점선으로 결과영상을 표현했습니다.