영상정보처리 4주차 과제 템플리트

- 점수: 10점 만점
- 이미지 경로 잘못 사용한 경우: -3
- 문제1: 5점
- 문제2: 5점

이름: 이현정

학번: 32203660

```
from google.colab import drive
drive.mount('/gdrive')

%cd /gdrive/MyDrive/ImageProcClass/Notebook-Week4
```

```
Drive already mounted at /gdrive; to attempt to forcibly remount, call drive.mount("/gdrive",
/gdrive/MyDrive/ImageProcClass/Notebook-Week4
```

Souce image 는 다음의 image_path 를 변경하지 말고 이용할 것. logo.png 는 opencv_logo.png 와 동일 이미지이므로 해당 이미지가 없는 경우, 복사하여 이름을 변경하여 사용할 것. 경로가 다른 경우 감점 -3

```
image_path = '../Dongkeun-OpenCV-ImgData/logo.png'
```

문제 1: 부분 이미지를 이용한 이미지 생성

1. 위의 이미지 경로를 이용하여 이미지를 컬러 이미지 org_image 로 읽기
2. 읽은 이미지와 동일 크기의 컬러 이미지를 new_image1 로 만들기
3. new_image1 를 x 축 방향으로 4등분하고 왼쪽부터 subimage 1, 2, 3, 4 라고 할때 다음과 같이 new_image1을 구성하기
    - x 축값이 4등분하여 4개의 영역이 동일 크기가 되지 않는 경우, 맨 오른쪽 영역에는 남는 크기 배분
    - subimage 1: 동일 영역에 해당하는 org_image 부분 복사하기
    - subimage 2: 동일 영역에 해당하는 org_image 부분에서 Red 성분만 복사하고, green/blue 부분은 0으로 하여 subimage 2에 채워 넣을 것
    - subimage 3: 동일 영역에 해당하는 org_image 부분에서 Blue 성분만 복사하고, green/red 부분은 0으로 하여 subimage 3에 채워 넣을 것
    - subimage 4: 동일 영역에 해당하는 org_image 부분에서 Green 성분만 복사하고, blue/red 부분은 0으로 하여 subimage 4에 채워 넣을 것
4. new_image1 을 디스플레이하기

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# 1. 컬러 이미지 읽기
org_image = cv2.imread(image_path)

# 2. 컬러 이미지 만들기
new_image1 = np.zeros(org_image.shape, dtype="uint8")

# 3. new_image 구성하기 : x축 값 4등분하여 4개 영역 분할
x_part = org_image.shape[1]//4
subimage1 = org_image[:,:x_part]

# 3-2. org_image에서 Red 성분만 복사
shp1 = org_image[:,x_part:x_part*2].shape # subimage2의 shape가 될 값
subimage2 = np.zeros(shp1, dtype="uint8") # 빈 이미지 생성
subimage2[:,:,2] = org_image[:,x_part:x_part*2,2] # BGR 중 Red 성분만 복사

# 3-3. org_image 부분에서 Blue 성분만 복사
shp2 = org_image[:,x_part*2:x_part*3].shape # subimage3의 shape가 될 값
subimage3 = np.zeros(shp2, dtype="uint8") # 빈 이미지 생성
subimage3[:,:,0] = org_image[:,x_part*2:x_part*3,0] # BGR 중 Blue 성분만 복사

# 3-4. org_image 부분에서 Green 성분만 복사
shp3 = org_image[:,x_part*3:].shape # subimage4의 shape가 될 값
subimage4 = np.zeros(shp3, dtype="uint8") # 빈 이미지 생성
subimage4[:,:,1] = org_image[:,x_part*3:,1] # BGR 중 Green 성분만 복사

new_image1[:,:x_part] = subimage1
new_image1[:,x_part:x_part*2] = subimage2
new_image1[:,x_part*2:x_part*3] = subimage3
new_image1[:,x_part*3:] = subimage4

# bgr -> rgb
new_image1 = new_image1[:,:,::-1]

# 4. new_image1 을 디스플레이
plt.imshow(new_image1)
plt.show()
```

문제 2: 식을 이용한 grayscale 이미지 만들기

1. org_image와 같은 크기의 공백 grayscale 이미지 new_image2 만들기
2. org_image 의 각 화소를 접근하여 강의에서 설명한 공식을 사용하여 grayscale 값으로 변환하여 새로운 그레이스케일 이미지 new_image2 에 저장하기
3. new_image2 디스플레이하기

[참조] https://stackoverflow.com/questions/17615963/standard-rgb-to-grayscale-conversion

```
# 1. grayscale 이미지 new_image2
new_image2 = org_image.copy()
# new_img2 = np.zeros(org_image.shape, dtype="uint8") -> 같은 크기 공백 이미지 생성
# new_image2 = cv2.cvtColor(new_img2, cv2.COLOR_BGR2GRAY) -> 공백 이미지를 grayscale 이미지로 변경

# 2. RGB -> gray
gray_channel = []
for i in range(new_image2.shape[0]):
  values = []
  for j in range(new_image2.shape[1]):
    value = 0.114*new_image2[i,j,0] + 0.587*new_image2[i,j,1] + 0.299*new_image2[i,j,2]
    values.append(value)
  gray_channel.append(values)

new_image2[:,:,0] = np.array(gray_channel)
new_image2 = new_image2[:,:,0]
print(new_image2.shape)

# 3. new_image2 디스플레이
plt.imshow(new_image2, cmap='gray')
plt.show()
```

과제 : 파이썬 파라미터 3.4.5 정리

```
def minimum(*n): # 파라미터 값을 tuple
  print('n :', n)
  print('type =', type(n))

  if n:
    mn = n[0]
    for value in n[1:]:
      if value < mn:
        mn = value
    print('-> mn:', mn)
    print()

minimum(1,3,-7,9)
# print : -7

minimum((1,3,-7,9))
# n의 원소는 tuple형태인 1개

minimum()
# n = ()
# print
```

```
n : (1, 3, -7, 9)
type = <class 'tuple'>
-> mn: -7

n : ((1, 3, -7, 9),)
type = <class 'tuple'>
-> mn: (1, 3, -7, 9)

n : ()
type = <class 'tuple'>
```

```
def func(**kwargs):
  print(kwargs)

# 받은 인자가 dict로 처리됨
func(a=1,b=42)

# dict형태로 전달하는 경우에는 ** 사용
func(**{'a':1,'b':42})
func(**dict(a=1,b=42))
```

```
{'a': 1, 'b': 42}
{'a': 1, 'b': 42}
{'a': 1, 'b': 42}
```

```
def kwo(*a, c):
  print(a, c)

kwo(1,2,3, c=7)
kwo(c=4)
# *는 값이 전달이 없어도 됨, 그러나 c는 반드시 필요
```

```
def kwo2(a, b=42, *, c):
  print(a,b,c)

# 필수적인 부분부터 처리
kwo2(3, b=7, c=99) # a=3
kwo2(3, c=13) # a=3
# kwo2(3, 23) : c 없으면 오류
```

```
(1, 2, 3) 7
() 4
3 7 99
3 42 13
```