KWANGWOON UNIVERSITY

Python NumPy Library

2020.11.09

Seoung-Jun Oh (sjoh@kw.ac.kr)
Jun-Young Sung (jysung13mmlab@kw.ac.kr)
Gi-Su Hwang (kisu031@kw.ac.kr)

Multimedia LAB

VIA-Multimedia Center, Kwangwoon University

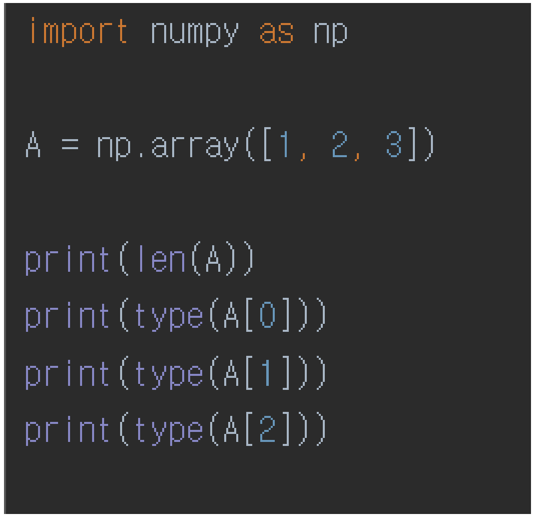
Contents

- NumPy
- Assignment
  - Color Conversion

NumPy

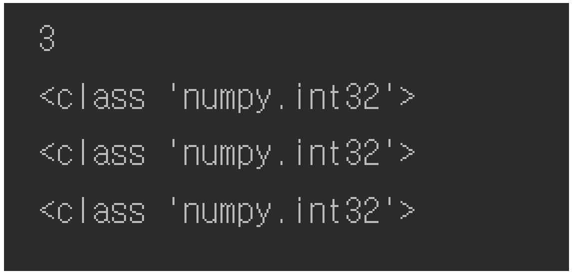
❖ Array

- array
- len
- type



```python
import numpy as np

A = np.array([1, 2, 3])

print(len(A))
print(type(A[0]))
print(type(A[1]))
print(type(A[2]))
```

CODE

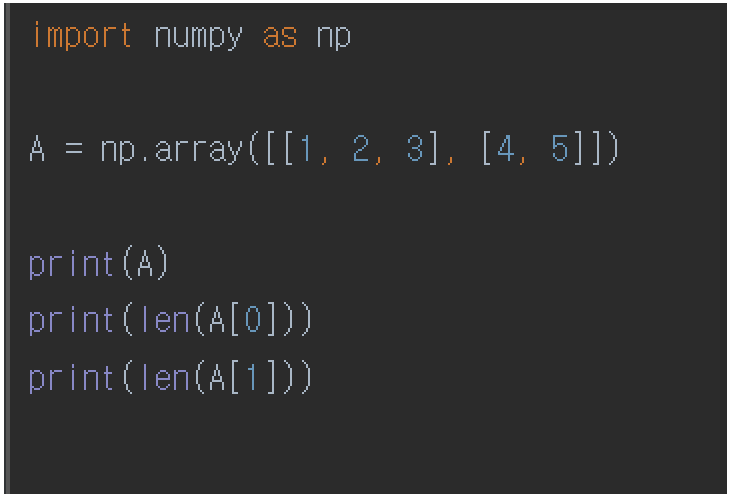


```
3
<class 'numpy.int32'>
<class 'numpy.int32'>
<class 'numpy.int32'>
```

Result

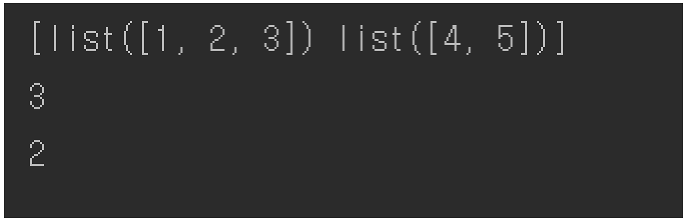
NumPy

- Array
  - array
  - len
  - type



```
import numpy as np

A = np.array([[1, 2, 3], [4, 5]])

print(A)
print(len(A[0]))
print(len(A[1]))
```

CODE



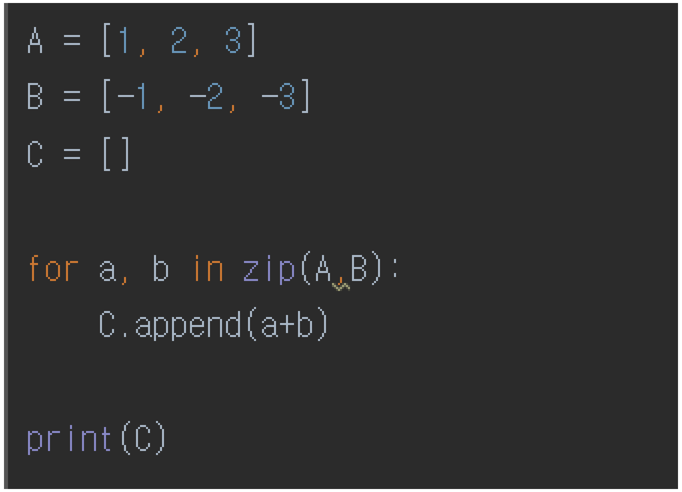
```
[list([1, 2, 3]) list([4, 5])]
3
2
```

Result

NumPy

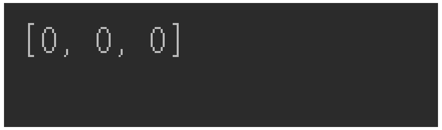
❖ List

- for
- append



```
A = [1, 2, 3]
B = [-1, -2, -3]
C = []

for a, b in zip(A,B):
    C.append(a+b)

print(C)
```

CODE



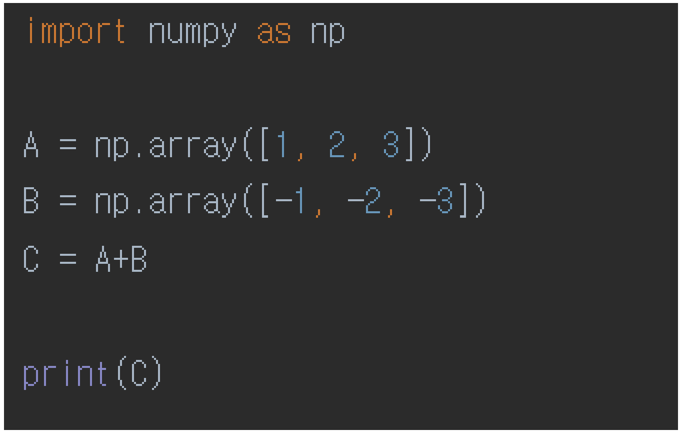
```
[0, 0, 0]
```

Result

NumPy

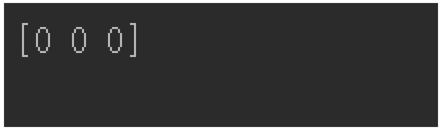
❖ Array

- A+B



```python
import numpy as np

A = np.array([1, 2, 3])
B = np.array([-1, -2, -3])
C = A+B

print(C)
```

CODE

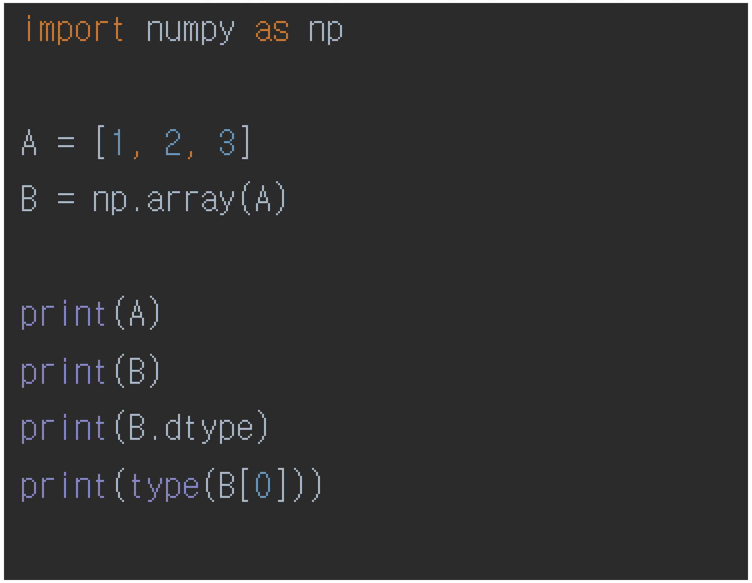


```
[0 0 0]
```

Result

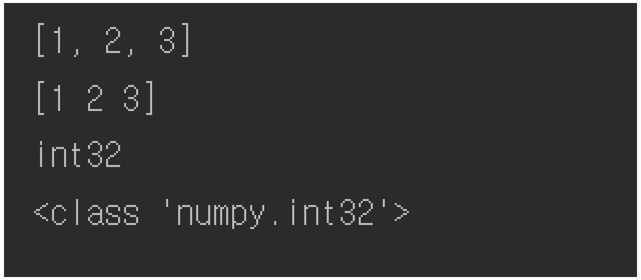
NumPy

❖ List to Array



```
import numpy as np

A = [1, 2, 3]
B = np.array(A)

print(A)
print(B)
print(B.dtype)
print(type(B[0]))
```

CODE



```
[1, 2, 3]
[1 2 3]
int32
<class 'numpy.int32'>
```

Result

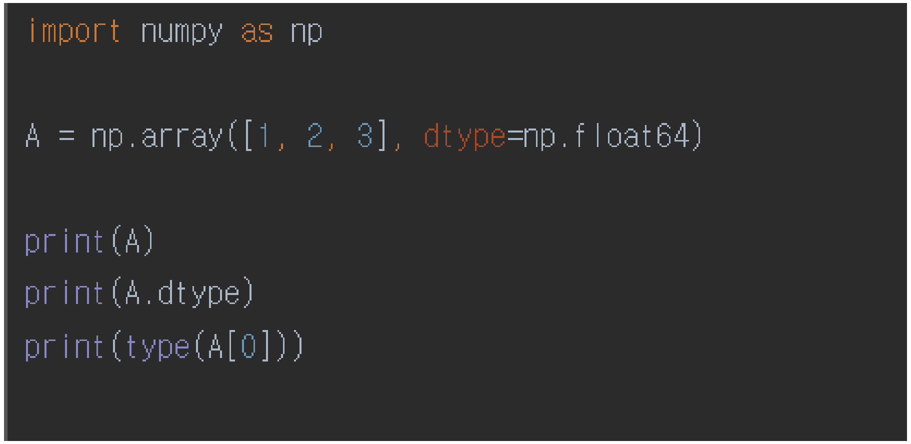
NumPy

❖ Data type

데이터 형	설명
int8, int16, int32, int64	부호가 있는 [8, 16, 32, 64]비트 정수
uint8, uint16, uint32, uint64	부호가 없는 [8, 16, 32, 64]비트 정수
float16,, float32, float64, float128	[16, 32, 64, 128]비트 실수
complex64, complex128, complex256	[64, 128, 256]비트 복소수
bool	True 또는 False
object	Python 오브젝트 형
string_	문자열
unicode_	유니코드 문자열

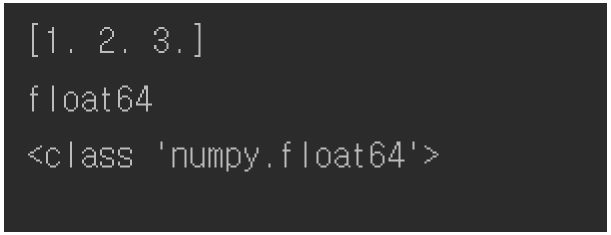
NumPy

- Array
  - dtype



```python
import numpy as np

A = np.array([1, 2, 3], dtype=np.float64)

print(A)
print(A.dtype)
print(type(A[0]))
```

CODE



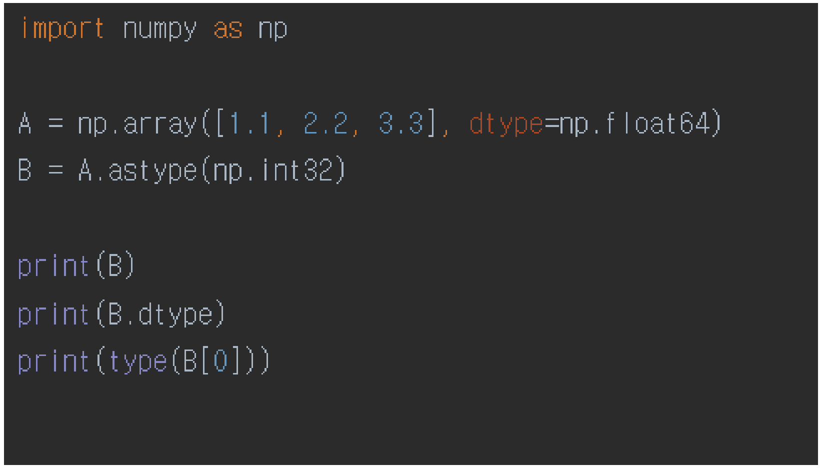
```
[1. 2. 3.]
float64
<class 'numpy.float64'>
```

Result

NumPy

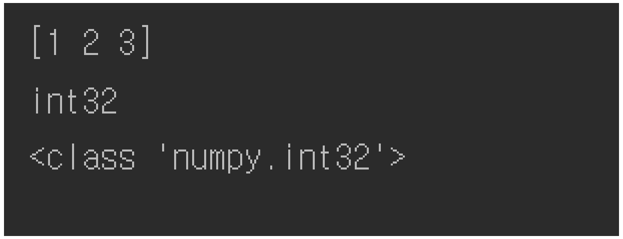
❖ Array

- astype



```
import numpy as np

A = np.array([1.1, 2.2, 3.3], dtype=np.float64)
B = A.astype(np.int32)

print(B)
print(B.dtype)
print(type(B[0]))
```

CODE



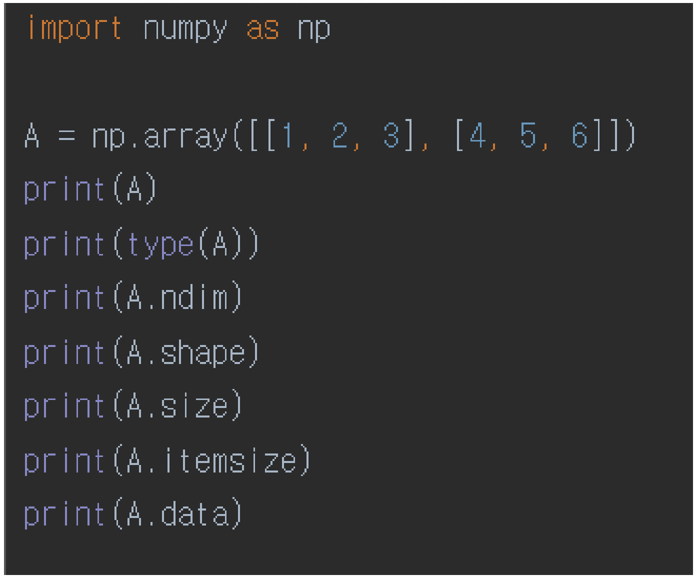
```
[1 2 3]
int32
<class 'numpy.int32'>
```

Result

NumPy

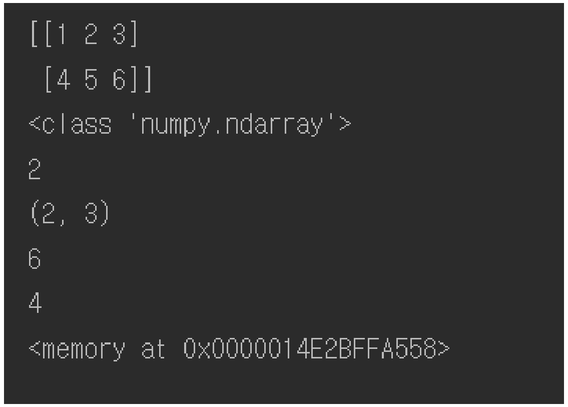
❖ Array

- type
- ndim
- shape
- size
- itemsize
- data



```
import numpy as np

A = np.array([[1, 2, 3], [4, 5, 6]])
print(A)
print(type(A))
print(A.ndim)
print(A.shape)
print(A.size)
print(A.itemsize)
print(A.data)
```

CODE

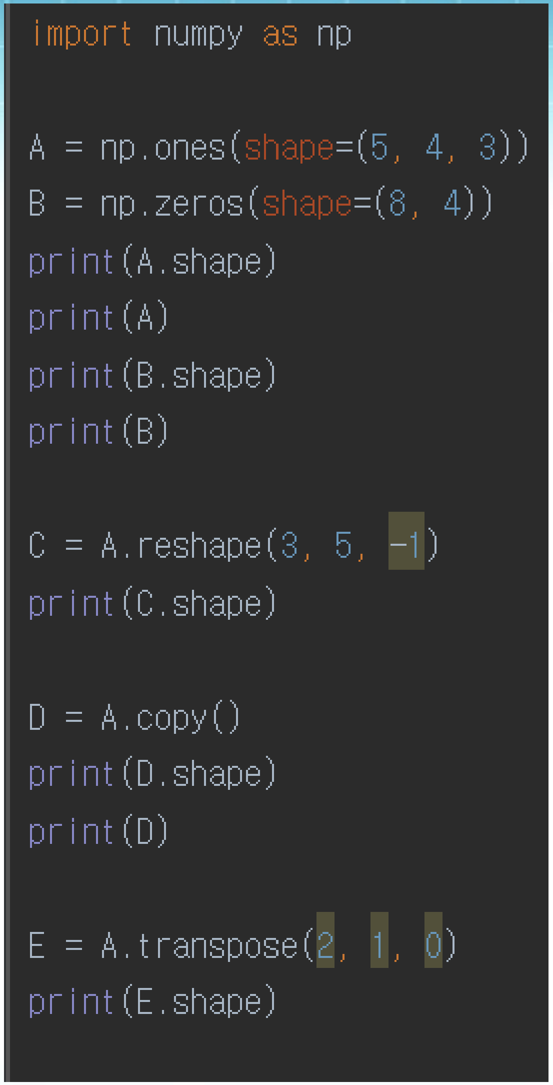


```
[[1 2 3]
 [4 5 6]]
<class 'numpy.ndarray'>
2
(2, 3)
6
4
<memory at 0x0000014E2BFFA558>
```

Result

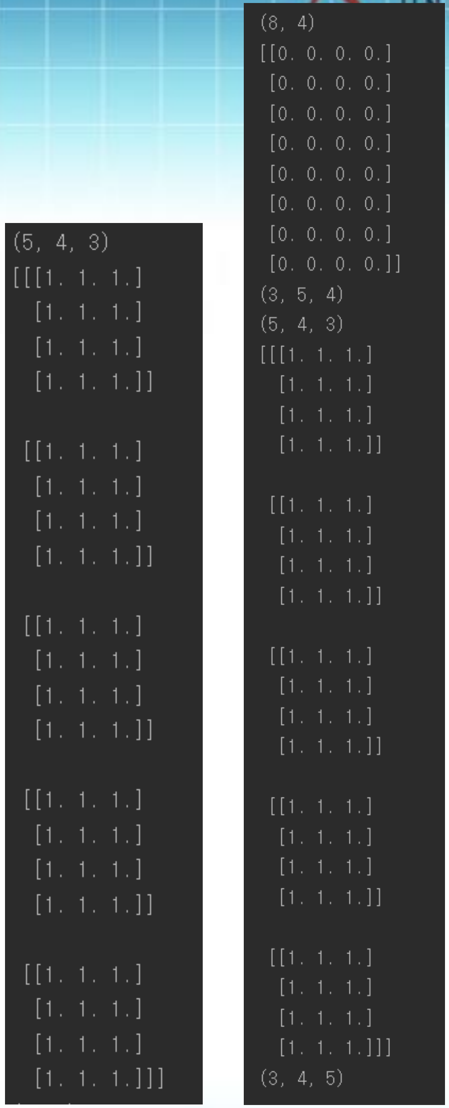
NumPy

- Array
  - ones
  - zeros
  - reshape
  - copy
  - transpose



```python
import numpy as np

A = np.ones(shape=(5, 4, 3))
B = np.zeros(shape=(8, 4))
print(A.shape)
print(A)
print(B.shape)
print(B)

C = A.reshape(3, 5, -1)
print(C.shape)

D = A.copy()
print(D.shape)
print(D)

E = A.transpose(2, 1, 0)
print(E.shape)
```

CODE



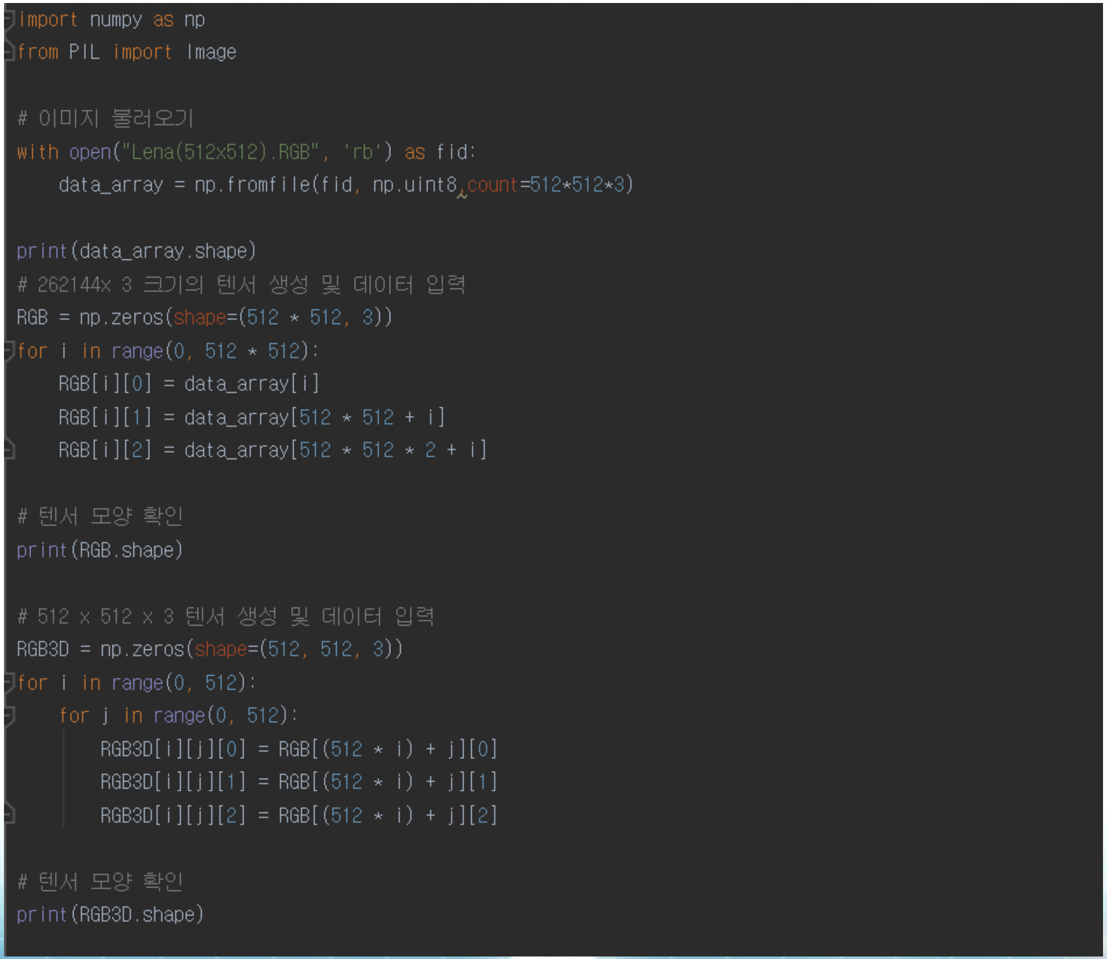
```
(5, 4, 3)
[[[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]]
```

```
(8, 4)
[[0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]]
(3, 5, 4)
(5, 4, 3)
[[[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]]
(3, 4, 5)
```

Result

Assignment

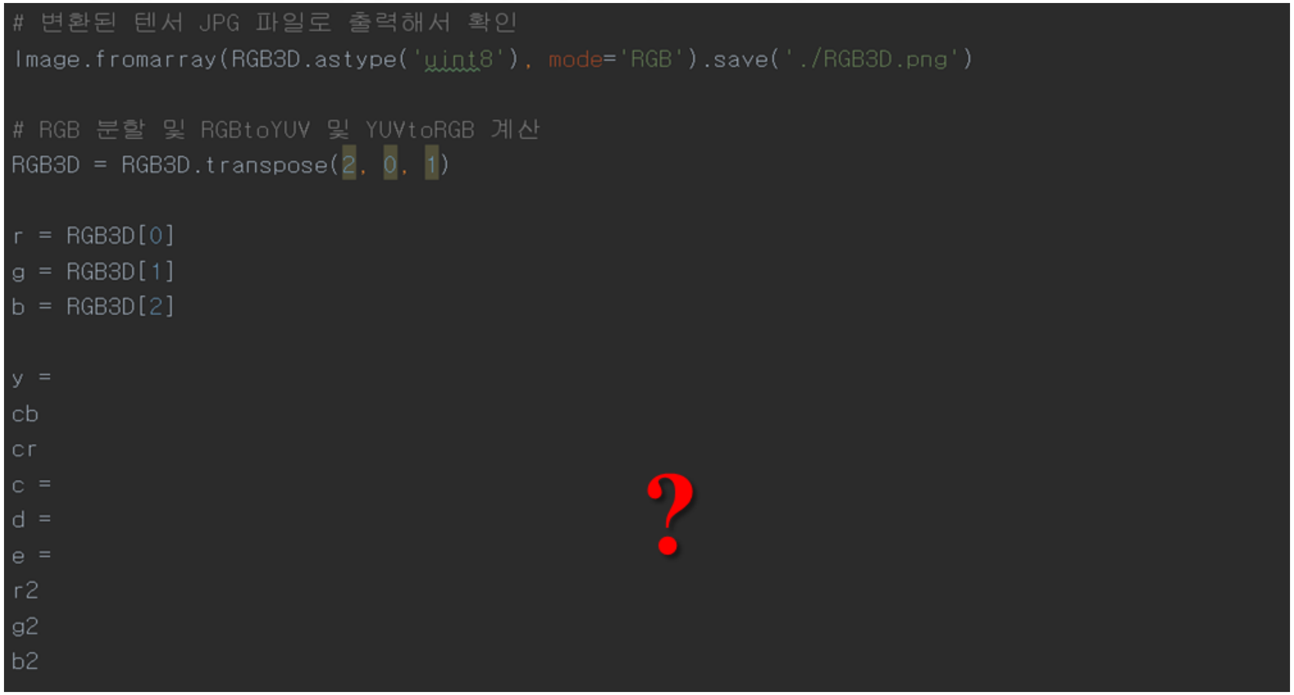
- Color Conversion



```
import numpy as np
from PIL import Image

# 이미지 불러오기
with open("Lena(512x512).RGB", 'rb') as fid:
    data_array = np.fromfile(fid, np.uint8,count=512*512*3)

print(data_array.shape)
# 262144x 3 크기의 텐서 생성 및 데이터 입력
RGB = np.zeros(shape=(512 * 512, 3))
for i in range(0, 512 * 512):
    RGB[i][0] = data_array[i]
    RGB[i][1] = data_array[512 * 512 + i]
    RGB[i][2] = data_array[512 * 512 * 2 + i]

# 텐서 모양 확인
print(RGB.shape)

# 512 x 512 x 3 텐서 생성 및 데이터 입력
RGB3D = np.zeros(shape=(512, 512, 3))
for i in range(0, 512):
    for j in range(0, 512):
        RGB3D[i][j][0] = RGB[(512 * i) + j][0]
        RGB3D[i][j][1] = RGB[(512 * i) + j][1]
        RGB3D[i][j][2] = RGB[(512 * i) + j][2]

# 텐서 모양 확인
print(RGB3D.shape)
```

Assignment

- Color Conversion

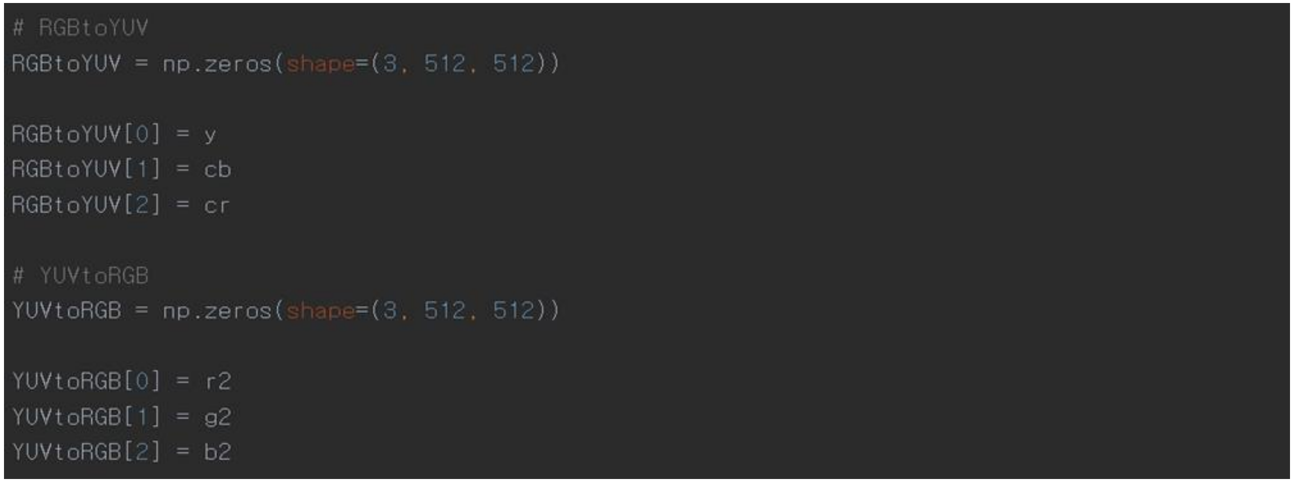


```
# 변환된 텐서 JPG 파일로 출력해서 확인
Image.fromarray(RGB3D.astype('uint8'), mode='RGB').save('./RGB3D.png')

# RGB 분할 및 RGBtoYUV 및 YUVtoRGB 계산
RGB3D = RGB3D.transpose(2, 0, 1)

r = RGB3D[0]
g = RGB3D[1]
b = RGB3D[2]

y =
cb
cr
c =
d =
e =
r2
g2
b2
```

?

Assignment

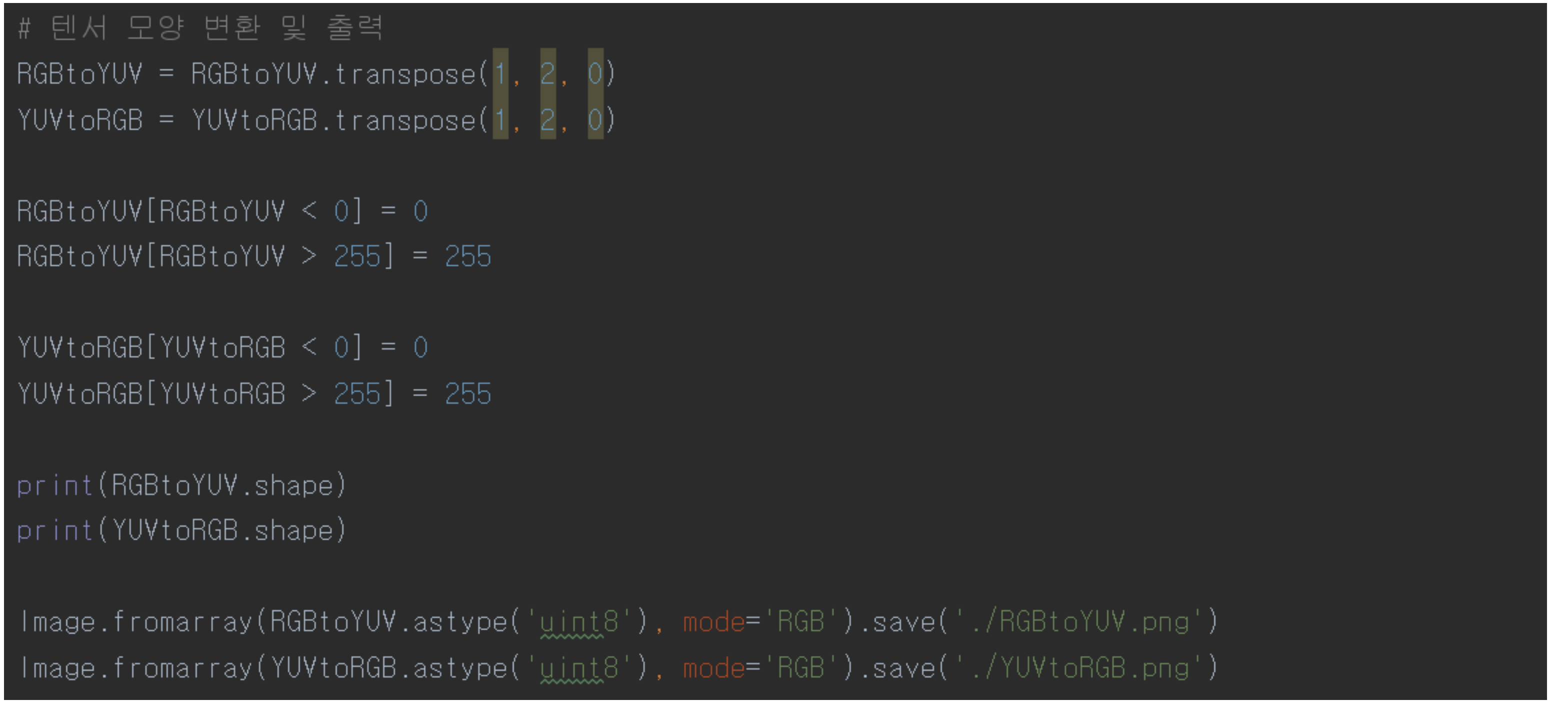
- Color Conversion



```
# RGBtoYUV
RGBtoYUV = np.zeros(shape=(3, 512, 512))

RGBtoYUV[0] = y
RGBtoYUV[1] = cb
RGBtoYUV[2] = cr

# YUVtoRGB
YUVtoRGB = np.zeros(shape=(3, 512, 512))

YUVtoRGB[0] = r2
YUVtoRGB[1] = g2
YUVtoRGB[2] = b2
```

Assignment

❖ Color Conversion



```
# 텐서 모양 변환 및 출력
RGBtoYUV = RGBtoYUV.transpose(1, 2, 0)
YUVtoRGB = YUVtoRGB.transpose(1, 2, 0)

RGBtoYUV[RGBtoYUV < 0] = 0
RGBtoYUV[RGBtoYUV > 255] = 255

YUVtoRGB[YUVtoRGB < 0] = 0
YUVtoRGB[YUVtoRGB > 255] = 255

print(RGBtoYUV.shape)
print(YUVtoRGB.shape)

Image.fromarray(RGBtoYUV.astype('uint8'), mode='RGB').save('./RGBtoYUV.png')
Image.fromarray(YUVtoRGB.astype('uint8'), mode='RGB').save('./YUVtoRGB.png')
```

Assignment

- Color Conversion