Numpy

https://youtu.be/bG6GRuUtCDg

Numpy

- 넘파이(Numpy)는 파이썬에서 벡터, 행렬 등 수치 연산을 수행하는 선형대수 라이브러리.
- 선형대수 관련 수치 연산을 지원하고 내부적으로 C로 구현되어 있어 연산이 빠른 속도로 수행됨.
- array라는 단위로 데이터를 관리하고, 행렬과 같은 개념이다.

Numpy (array 생성)

import numpy as np

```
[2]   1 a = [1, 2, 3, 4]
      2 print(a)
```

```
[1, 2, 3, 4]
```

```
[4]   1 b = np.array(a)
      2 print(b)
```

```
[1 2 3 4]
```

```
[5]   1 c = np.array([1,2,3,4])
      2 # c = np.array(1,2,3,4)  잘못된방법
      3 print(c)
```

```
[1 2 3 4]
```

```
[6]   1 d = np.arange(1,5,1)
      2 # arange(a, b, c) a이상 b미만! c만큼 차이나는 array를 만듬
      3 print(d)
```

```
[1 2 3 4]
```

Numpy (arange / reshape)

```
[13]  1 a = np.arange(15)
      2 print(a); print()
      3
      4 a = a.reshape(3, 5)
      5 print(a); print()
      6
      7 a = np.arange(15).reshape(3,5)
      8 print(a)
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14]

[[ 0  1  2  3  4]
 [ 5  6  7  8  9]
 [10 11 12 13 14]]

[[ 0  1  2  3  4]
 [ 5  6  7  8  9]
 [10 11 12 13 14]]
```

Numpy (속성값 확인)

```
[ ]  1 print("a.shape          =",a.shape)
     2 # ndarray.shape : 배열의 각 축(axis)의 크기
     3
     4 print("a.ndim           =",a.ndim)
     5 # ndarray.ndim : 축의 개수(Dimension)
     6
     7 print("a.dtype          =",a.dtype)
     8 # ndarray.dtype : 각 요소(Element)의 타입
     9
    10 print("a.itemsize       =",a.itemsize)
    11 # ndarray.itemsize : 각 요소(Element) 타입의 byte 크기
    12
    13 print("a.size           =",a.size)
    14 # ndarray.size : 전체 요소(Element)의 개수
    15
    16 print("type(a)          =",type(a))
```

```
a.shape          = (3, 5)
a.ndim           = 2
a.dtype          = int64
a.itemsize       = 8
a.size           = 15
type(a)          = <class 'numpy.ndarray'>
```

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14

shape = (3, 5)

ndim = 2

size = 15

a[3][5]

Numpy (zeros, ones, empty)

```
1 zero = np.zeros((3,5))
2 print(zero); print()
3
4 ones = np.ones((3,5), dtype=np.int32)
5 print(ones); print()
6
7 empty = np.empty((3,5))
8 print(empty)
```

```
[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]

[[1 1 1 1 1]
 [1 1 1 1 1]
 [1 1 1 1 1]]

[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
```

Numpy (사칙연산)

```
[37]  1 a = np.ones((4,4), dtype=np.int32)
      2 print(a); print()
      3
      4 b = np.arange(16).reshape(4,4)
      5 print(b)
```

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

[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]
 [12 13 14 15]]
```

```
[38]  1 print(a+b)
```

```
[[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]
 [13 14 15 16]]
```

```
[39]  1 print(a-b)
```

```
[[  1   0  -1  -2]
 [ -3  -4  -5  -6]
 [ -7  -8  -9 -10]
 [-11 -12 -13 -14]]
```

```
[41]  1 print((a+b)*b)
```

```
[[  0   2   6  12]
 [ 20  30  42  56]
 [ 72  90 110 132]
 [156 182 210 240]]
```

```
[42]  1 print(a/b)
```

```
[[       inf 1.         0.5        0.33333333]
 [0.25       0.2        0.16666667 0.14285714]
 [0.125      0.11111111 0.1        0.09090909]
 [0.08333333 0.07692308 0.07142857 0.06666667]]
```

Numpy (사직연산 list와 다른점)

```
1 a = [1,1,1,1]
2 b = [0,1,2,3]
3 print(a+b); print()
4
5 na = np.array(a)
6 nb = np.array(b)
7 print(na+nb)
```

```
[1, 1, 1, 1, 0, 1, 2, 3]

[1 2 3 4]
```

list 에서는 다른 사칙연산이 작동하지 않음

```
1 print(np.hstack((na,nb))); print();
2 # axis = 1  기준으로 쌓음 / 가로로 쌓는다
3
4 print(np.vstack((na,nb)))
5 # axis = 0  기준으로 쌓음 / 세로로 쌓는다
```

```
[1 1 1 1 0 1 2 3]

[[1 1 1 1]
 [0 1 2 3]]
```

numpy에서 list의 더하기 역할
= .hstack

Numpy (브로드 캐스트)

브로드캐스트란, 서로 크기가 다른 array가 연산이 가능하게끔 하는 것.

```
[66]  1 a = np.arange(15).reshape(3,5)
      2 print(a); print()
      3
      4 b = np.array([2,2,2,2,2])
      5 print(b); print()
      6
      7 print(a.shape)
      8 print(b.shape)
```

```
[[ 0  1  2  3  4]
 [ 5  6  7  8  9]
 [10 11 12 13 14]]

[2 2 2 2 2]

(3, 5)
(5,)
```

```
1 print(a*b); print()
2 print(a*2); print()
3 print(a**2)
```

```
[[ 0  2  4  6  8]
 [10 12 14 16 18]
 [20 22 24 26 28]]

[[ 0  2  4  6  8]
 [10 12 14 16 18]
 [20 22 24 26 28]]

[[  0   1   4   9  16]
 [ 25  36  49  64  81]
 [100 121 144 169 196]]
```

Numpy (복사)

```
[84]  1 a = np.arange(10)
      2 print(a)
      3
      4 c = a
      5 print(c)
      6
      7 c[1] = 9
      8 print(a); print()
      9
     10 print(a is c)
     11 print(id(a))
     12 print(id(c))
```

```
[0 1 2 3 4 5 6 7 8 9]
[0 1 2 3 4 5 6 7 8 9]
[0 9 2 3 4 5 6 7 8 9]

True
139843895861936
139843895861936
```

```
1 c = c.reshape(2,5)
2 print(c); print()
3 print(a)
4 print(a is c); print()
5
6 a[1] = 2
7 print(c)
```

```
[[0 2 2 3 4]
 [5 6 7 8 9]]

[0 2 2 3 4 5 6 7 8 9]
False

[[0 2 2 3 4]
 [5 6 7 8 9]]
```

```
 1 a = np.arange(10)
 2 print(a)
 3
 4 c = a.copy()
 5 print(c)
 6
 7 c[1] = 2
 8 print(a); print()
 9
10 print(a is c)
11 print(id(a))
12 print(id(c))
```

```
[0 1 2 3 4 5 6 7 8 9]
[0 1 2 3 4 5 6 7 8 9]
[0 1 2 3 4 5 6 7 8 9]

False
139843896816784
139843896818224
```

Numpy (Array 인덱싱)

```
1 a = np.arange(10)
2 print(a); print()
3
4 print(a[7])
5 print(a[3:5])
6 print(a[:7:2])
7 print(a[4:])
8
9 # a[시작:끝:차이]
```

```
[0 1 2 3 4 5 6 7 8 9]

7
[3 4]
[0 2 4 6]
[4 5 6 7 8 9]
```

```
1 a = np.arange(20).reshape(4,5)
2 print(a); print()
3
4 print(a[0][0])
5 #print(a[0,0])
6 print(a[:2][:2])  # wrong
7 print()
8
9 print(a[:2 , :2])
```

```
[[ 0  1  2  3  4]
 [ 5  6  7  8  9]
 [10 11 12 13 14]
 [15 16 17 18 19]]

0
[[0 1 2 3 4]
 [5 6 7 8 9]]

[[0 1]
 [5 6]]
```

Numpy (Boolean Array)

```
[5]  1 a = np.arange(12).reshape(3,4)
     2 print(a); print()
     3
     4 b = a > 4
     5 print(b)
```

```
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]

[[False False False False]
 [False  True  True  True]
 [ True  True  True  True]]
```

```
 1 print(a[b]); print()
 2
 3 a[b] = 0
 4 print(a)
```

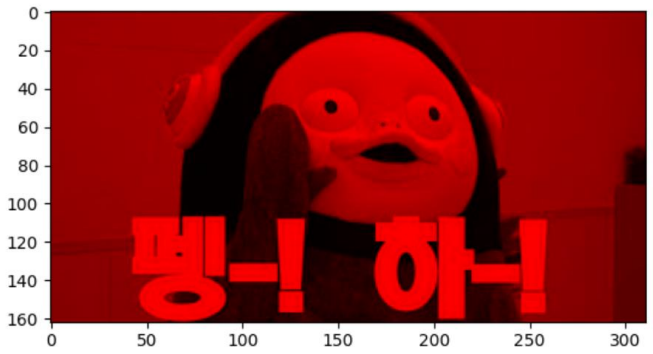
```
[ 5  6  7  8  9 10 11]

[[0 1 2 3]
 [4 0 0 0]
 [0 0 0 0]]
```

Numpy 이미지 실습

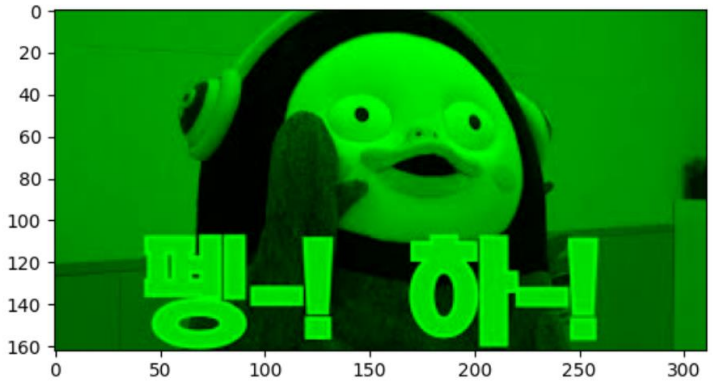
```python
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt

img = Image.open('./sample.jpeg')

npimage = np.array(img)

plt.imshow(npimage)
plt.show()
```


```
shape = (162, 311, 3)
```

Numpy 이미지 실습



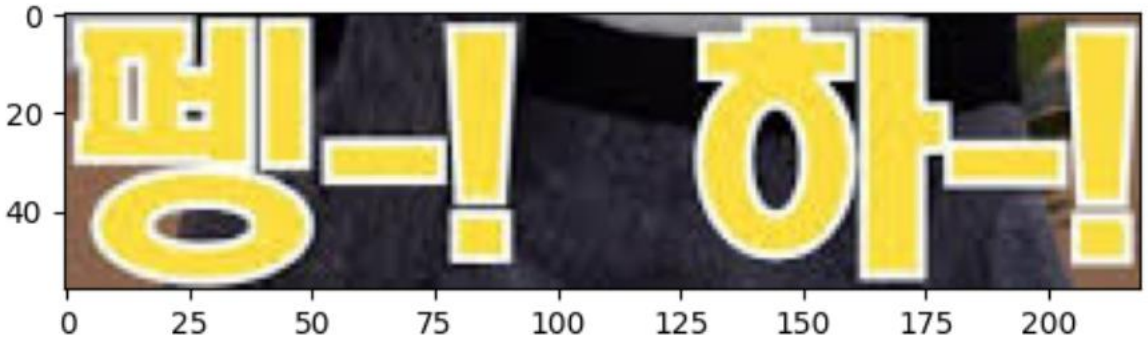

```
npimage[:, :, 1] = 0
npimage[:, :, 2] = 0
```




```
npimage[:, :, 0] = 0
npimage[:, :, 2] = 0
```


```
npimage[:, :, 0] = 0
npimage[:, :, 1] = 0
```

Numpy 이미지 실습




```
npimage_hi = npimage[106:162, 41:260, :]
```

Q & A