

파이선으로 배우는 데이터구조

Data Structures Learning with Python

김영훈

한양대학교 ERICA 인공지능학과

넘파이 다차원 배열의 사용 예

- ▶ 생성하기: `np.array()`

```
import numpy as np
sampleArray = np.array([[11, 22, 33], [44, 55, 66], [77, 88, 99]])
print(sampleArray)
```

```
[[11 22 33]
 [44 55 66]
 [77 88 99]]
```

넘파이 다차원 배열의 사용 예

- ▶ 축 방향 벡터 가져오기 (슬라이싱)

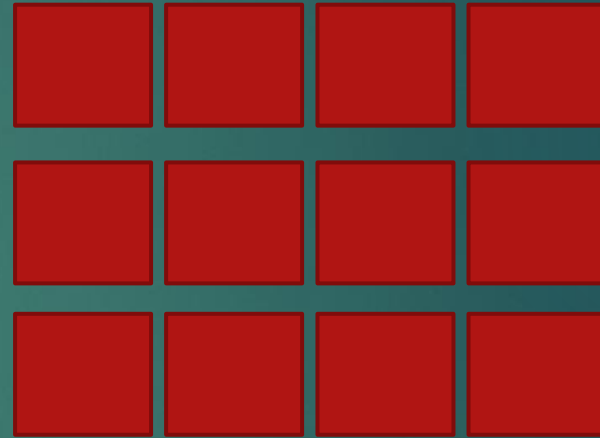
```
sampleArray = np.array([[11, 22, 33], [44, 55, 66], [77, 88, 99]])  
print(sampleArray[0, :])  
print(sampleArray[:, 1])  
print(sampleArray[:, -1])
```

```
[11 22 33]  
[22 55 88]  
[33 66 99]
```

다차원 배열?

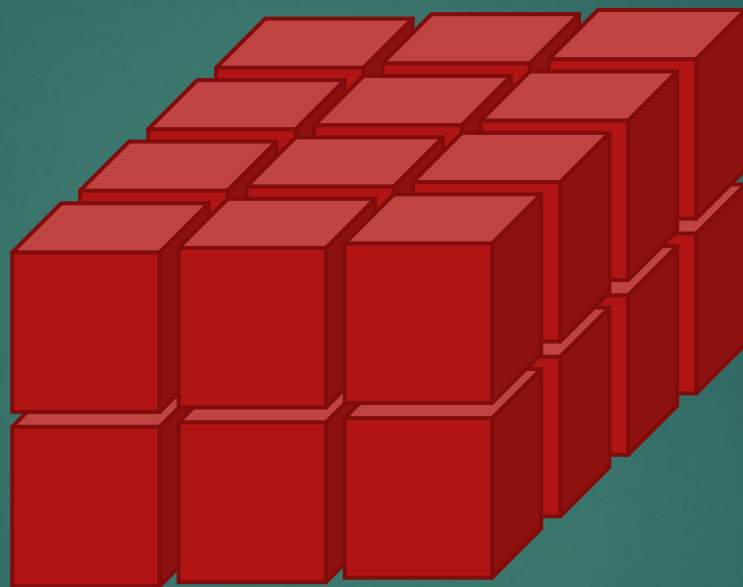


1D array with shape (4,)



2D array with shape (3, 4)

다차원 배열?



3D array with shape (2, 3, 4)

Python List vs numpy.ndarray

▶ Numpy.ndarray → fixed type

▶ E.g.,

▶ `a = [15, 'a']`

string

int64

Object value = 0x0000 0000 0000 000f,

▶ `a = np.array([15, 16], dtype='int32')` Object value = 0x0000 000f

int64


Python List vs numpy.ndarray

`a=np.array([1, 2, 3]):`

`a=` 



`a=[1, 3.2, 'abc']:`

`a=`  `]`



Element-wise Computation



List

```
a = [1, 2, 3]
```

```
b = [2, 4, 6]
```

```
a+b → error
```



Numpy
.ndarray

```
a = numpy.array([1, 2, 3])
```

```
b = numpy.array([2, 4, 6])
```

```
a+b → numpy.array([3, 6, 9])
```


Numpy 1.

- ▶ 4X2 정수 배열(16비트)을 작성하고 해당 특성을 출력하세요.
- ▶ 출력: Shape: (4, 2)
Dimensions: 2
Datatype: uint16

```
firstArray = numpy.empty([4,2], dtype = numpy.uint16)

print("Shape: ", firstArray.shape)
print("Dimension: ", firstArray.ndim)
print("Data type: ", firstArray.dtype)
```

Numpy 2.

- ▶ 제공된 NumPy 배열에서 모든 행에서 세 번째 열을 가져와 항목 배열을 반환하세요.
- ▶ 입력: `numpy.array([[11, 22, 33], [44, 55, 66], [77, 88, 99]])`
- ▶ 출력: `[33 66 99]`

```
sampleArray = numpy.array([[11, 22, 33], [44, 55, 66], [77, 88, 99]])  
  
newArray = sampleArray[:,2]  
print(newArray)
```

Numpy 3.

- ▶ 홀수 행과 짝수 열의 배열 반환

```
sampleArray = np.array([
    [3 ,6, 9, 12],
    [15 ,18, 21, 24],
    [27 ,30, 33, 36],
    [39 ,42, 45, 48],
    [51 ,54, 57, 60]])

print(sampleArray[::2, 1::2])
```

Numpy 4.

- ▶ 넘파이 배열의 부분 인덱스로 슬라이싱

```
sampleArray = np.array([[34,43,73],[82,22,12],[53,94,66]])  
print(sampleArray[:-1])
```

Numpy 4.

- ▶ 넘파이 배열의 부분 인덱스로 슬라이싱

```
sampleArray = np.array([[34,43,73],[82,22,12],[53,94,66]])  
print(sampleArray[:, [0,2]])
```

Numpy 5.

- ▶ 두 개의 NumPy 배열의 덧셈 및 곱셈 계산

```
arrayOne = ny.array([[5, 6, 9], [21 ,18, 27]])  
arrayTwo = ny.array([[15 ,33, 24], [4 ,7, 1]])  
  
print(arrayOne + arrayTwo)  
print(arrayOne * arrayTwo)
```

Numpy 6.

- ▶ 축 0 (같은 열번호를 갖는 혹은 **행번호를 증가시키며**) 에서 최대값을 출력하고
- ▶ 축 1 (같은 행번호를 갖는 혹은 **열번호를 증가시키며**) 에서 최소값을 출력

```
sampleArray = np.array([[34,43,73],[82,22,12],[53,94,66]])  
print(np.max(sampleArray, axis=0))
```

```
sampleArray = np.array([[34,43,73],[82,22,12],[53,94,66]])  
print(np.min(sampleArray, axis=1))
```

Numpy 7.

- ▶ 축 0 (행렬 번호를 증가시키며) 에서 합산을 출력

```
sampleArray = np.array([
    [[3,4,3],
     [8,2,2],
     [5,4,6]],
    [[1,2,2],
     [4,2,2],
     [3,9,6]]
])
print(np.sum(sampleArray, axis=0))
```


Numpy 8.

- ▶ 축 2 (마지막 축의 번호를 증가시키며) 에서 합산을 출력

```
sampleArray = np.array([
    [[3,4,3],
     [8,2,2],
     [5,4,6]],
    [[1,2,2],
     [4,2,2],
     [3,9,6]]
])
print(np.sum(sampleArray, axis=2))
```

Numpy 9.

- ▶ 축 1 (두번째 축의 번호를 증가시키며) 에서 합산을 출력

```
sampleArray = np.array([
    [[3,4,3],
     [8,2,2],
     [5,4,6]],
    [[1,2,2],
     [4,2,2],
     [3,9,6]]
])
print(np.sum(sampleArray, axis=1))
```

Numpy 10.

- ▶ True / False를 이용한 인덱스 선택

```
sampleArray = np.array([
    [[3,4,3],
     [8,2,2],
     [5,4,6]],
    [[1,2,2],
     [4,2,2],
     [3,9,6]]
])
print(sampleArray[[True, False], :, :])
```

Numpy 11.

- ▶ True / False를 이용한 인덱스 선택

```
sampleArray = np.array([
    [[3,4,3],
     [8,2,2],
     [5,4,6]],
    [[1,2,2],
     [4,2,2],
     [3,9,6]]
])
print(sampleArray[:, [True, False, True], :])
```

Numpy 12.

- ▶ 비교연산 – 각 행의 첫번째 열이 5보다 큰가?

```
sampleArray = np.array([
    [3,4,3],
    [8,2,2],
    [5,4,6]])
print(sampleArray[:, 0] > 4)
```

Numpy 13.

- ▶ 첫번째 열 값이 5보다 큰 행만 출력하기

```
sampleArray = np.array([
    [3,4,3],
    [8,2,2],
    [5,4,6]])
print(sampleArray[sampleArray[:, 0] > 4, :])
```

Numpy 14.

- ▶ 다음 넘파이 배열에서 두번째 열과 세번째 열의 합이 100보다 큰 행만 출력

```
sampleArray = np.array([[34,43,73,54],[82,22,12,22],[53,94,66,37]])
```

?

Numpy 15.

- ▶ 다음과 같이 출력되도록 슬라이싱
- ▶ 출력: $\begin{bmatrix} 3 & 3 \\ 5 & 6 \end{bmatrix}$

```
sampleArray = np.array([  
    [3,4,3],  
    [8,2,2],  
    [5,4,6]])
```

?

Numpy 16.

- ▶ 홀수 인덱스 열만 출력

```
sampleArray = np.array(range(100)).reshape((10, 10))
```

?

```
[[ 1  3  5  7  9]
 [11 13 15 17 19]
 [21 23 25 27 29]
 [31 33 35 37 39]
 [41 43 45 47 49]
 [51 53 55 57 59]
 [61 63 65 67 69]
 [71 73 75 77 79]
 [81 83 85 87 89]
 [91 93 95 97 99]]
```

Numpy 17.

- ▶ 다음은 네 학생의 국영수 성적이다. 넘파이 배열로 표현하고 총점을 계산하시오.

34	43	73
82	22	12
53	94	66
42	53	33

?

[169 159 151]