■ 배열 내부 연산

● 같은 위치에 있는 요소들의 연산

```
a = np.arange(1, 7, dtype=np.float).reshape(2, 3)
b = np.arange(11, 17, dtype=np.float).reshape(2, 3)

print(a)
print(b)
```

```
[[1. 2. 3.]
 [4. 5. 6.]]
[[11. 12. 13.]
 [14. 15. 16.]]
```

```
a + b  # 더하기
```

```
array([[12., 14., 16.],
       [18., 20., 22.]])
```

```
a - b  # 빼기
```

```
array([[-10., -10., -10.],
       [-10., -10., -10.]])
```

■ 배열 내부 연산

● 같은 위치에 있는 요소들의 연산

```
a * b  # 곱하기
```

```
array([[11., 24., 39.],
       [56., 75., 96.]])
```

```
a / b  # 나누기
```

```
array([[0.09090909, 0.16666667, 0.23076923],
       [0.28571429, 0.33333333, 0.375     ]])
```

```
b // a  # 몫
```

```
array([[11.,  6.,  4.],
       [ 3.,  3.,  2.]])
```

```
b % a  # 나머지
```

```
array([[0., 0., 1.],
       [2., 0., 4.]])
```

```
a ** b  # 제곱
```

```
array([[1.00000000e+00, 4.09600000e+03, 1.59432300e+06],
       [2.68435456e+08, 3.05175781e+10, 2.82110991e+12]])
```

■ 배열 내부 연산

● 같은 위치에 있는 요소들의 연산

```
d = np.arange(1, 7).reshape(2, 3)
e = np.arange(11, 17).reshape(2, 3)
f = np.arange(21, 27).reshape(2, 3)

print(d)
print(e)
print(f)
```

```
[[1 2 3]
 [4 5 6]]
[[11 12 13]
 [14 15 16]]
[[21 22 23]
 [24 25 26]]
```

```
d + e + f
```

```
array([[33, 36, 39],
       [42, 45, 48]])
```

■ 배열 내부 연산

● 행렬 곱셈 : dot()

```
g = np.arange(1, 7).reshape(2, 3)
h = np.arange(11, 17).reshape(3, 2)
```

```
g.dot(h)
```

```
array([[ 82,  88],
       [199, 214]])
```

1	2	3
4	5	6

×

11	12
13	14
15	16

=

82	88
199	214

- transpose 사용

```
g
```

```
array([[1, 2, 3],
       [4, 5, 6]])
```

×

```
g.T
```

```
array([[1, 4],
       [2, 5],
       [3, 6]])
```

=

```
g.dot(g.T)
```

```
array([[14, 32],
       [32, 77]])
```

■ 배열 내부 연산

● 행렬 곱셈 : dot()

- 1차원 배열에서의 dot() 사용

```
test_a = np.arange(1, 4)
test_b = np.arange(11, 14)
```

```
test_a * test_b
```

```
array([11, 24, 39])
```

```
test_a.dot(test_b)
```

```
74
```

■ Broadcasting

- shape이 다른 배열 간 연산 지원
- scalar – vector – matrix – tensor 모두 연산 가능

– scalar / vector

```
scalar = 10
vector = np.array([1, 2, 3])
```

```
scalar + vector
```

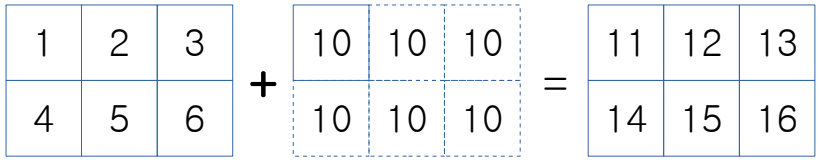
```
array([11, 12, 13])
```

– scalar / matrix

```
scalar = 10
matrix = np.array([[1, 2, 3], [4, 5, 6]])
```

```
scalar + matrix
```

```
array([[11, 12, 13],
       [14, 15, 16]])
```



■ Broadcasting

- ● shape이 다른 배열 간 연산 지원
- ● scalar – vector – matrix – tensor 모두 연산 가능
  - scalar / tensor

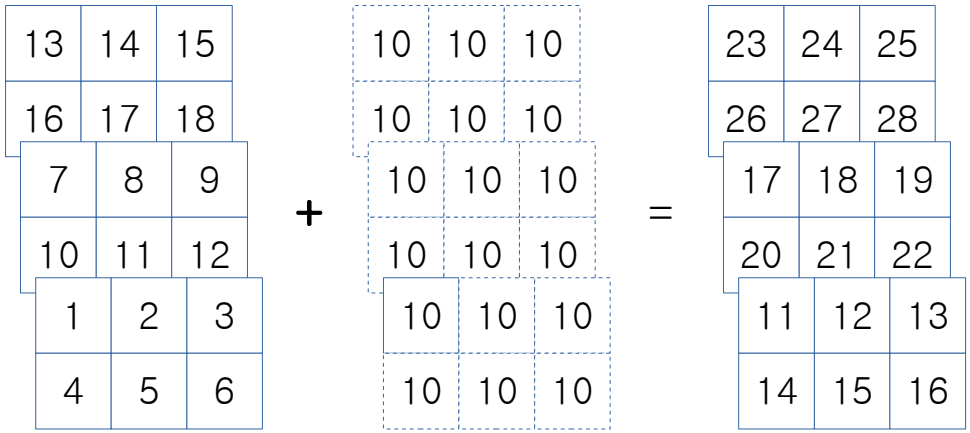
```
scalar = 10
tensor = np.array(
    [
        [
            [1, 2, 3], [4, 5, 6]
        ],
        [
            [7, 8, 9], [10, 11, 12]
        ],
        [
            [13, 14, 15], [16, 17, 18]
        ]
    ]
)
```

```
scalar + tensor
```

```
array([[[11, 12, 13],
        [14, 15, 16]],

       [[17, 18, 19],
        [20, 21, 22]],

       [[23, 24, 25],
        [26, 27, 28]]])
```



■ Broadcasting

- shape이 다른 배열 간 연산 지원
- scalar – vector – matrix – tensor 모두 연산 가능

– vector / matrix

```
matrix = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
vector = np.array([11, 12, 13])
```

```
matrix + vector
```

```
array([[12, 14, 16],
       [15, 17, 19],
       [18, 20, 22]])
```

1	2	3		11	12	13		12	14	16
4	5	6	+	11	12	13	=	15	17	19
7	8	9		11	12	13		18	20	22

– vector / vector

```
vector1 = np.array([[1], [2], [3]])
vector2 = np.array([11, 12, 13])
```

```
vector1 + vector2
```

```
array([[12, 13, 14],
       [13, 14, 15],
       [14, 15, 16]])
```

1	1	1		11	12	13		12	13	14
2	2	2	+	11	12	13	=	13	14	15
3	3	3		11	12	13		14	15	16