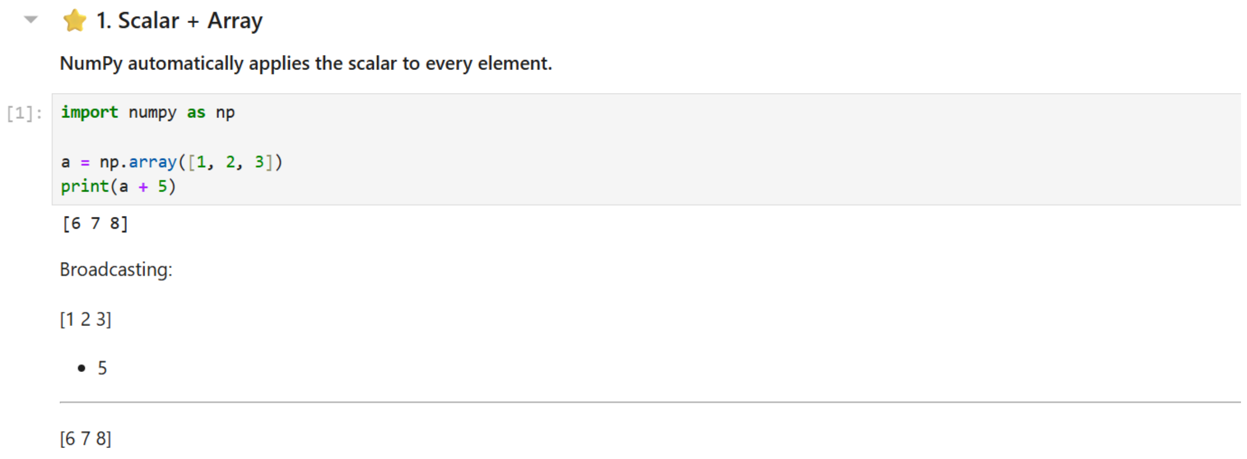
⭐ Broadcasting

- Broadcasting means NumPy **expands smaller arrays** to match the shape of the larger array and applies the operation **element-by-element** automatically, without loops.



⭐ 1. Scalar + Array

NumPy automatically applies the scalar to every element.

```
[1]: import numpy as np

     a = np.array([1, 2, 3])
     print(a + 5)
```

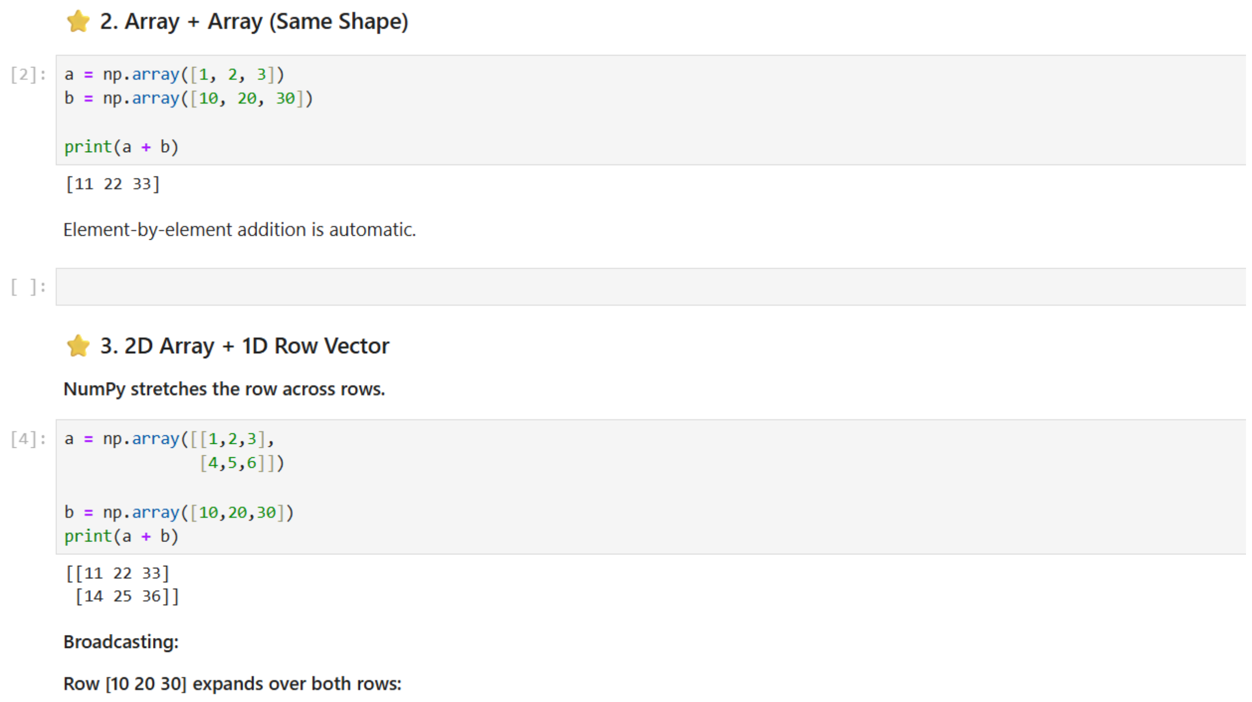
```
[6 7 8]
```

Broadcasting:

[1 2 3]

- 5

[6 7 8]



⭐ 2. Array + Array (Same Shape)

```
[2]: a = np.array([1, 2, 3])
     b = np.array([10, 20, 30])

     print(a + b)
```

```
[11 22 33]
```

Element-by-element addition is automatic.

```
[ ]:
```

⭐ 3. 2D Array + 1D Row Vector

NumPy stretches the row across rows.

```
[4]: a = np.array([[1,2,3],
                   [4,5,6]])

     b = np.array([10,20,30])
     print(a + b)
```

```
[[11 22 33]
 [14 25 36]]
```

Broadcasting:

Row [10 20 30] expands over both rows:

⭐ 4. 2D Array + 1D Column Vector

Column expands down.

[6]:

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

c = np.array([[10],
              [20]])

print(A + c)
```

```
[[11 12 13]
 [24 25 26]]
```

Broadcasting:

[[10], → [[10 10 10]

[20]] - [20 20 20]]

Then added to A.

[ ]:

```
```

⭐ 5. Mixing different shapes

[7]:

```
A = np.array([[1, 2, 3],
              [4, 5, 6],
              [7, 8, 9]])

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

print(A + b)
```

```
[[ 2  4  6]
 [ 5  7  9]
 [ 8 10 12]]
```

This works because b expands across each row.

[ ]:

```
```