In [1]:

```
from IPython.core.interactiveshell import InteractiveShell

InteractiveShell.ast_node_interactivity = "all"
```

In [2]:

```
import numpy as np
```

2-1

```
![2-1](2_1.jpg)
```

2-2

In [3]:

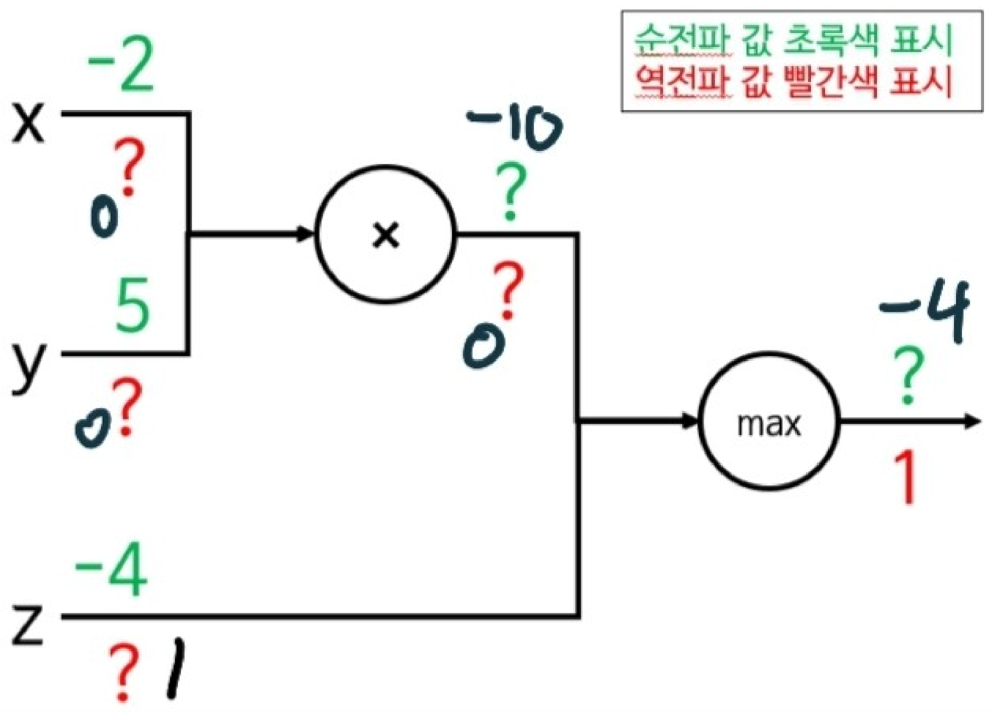
```
x , y, z = 2, 2, 2
f = 2*x*y + 3*(x**2)*z + 4*z # f 식

dfdx = 2*y + 6*x*z

print(dfdx)
```

```
28
```

3-1



3-2

In [4]:

```
x = -2; y = 5; z = -4;
p = x*y
f = max(p,z)
print(f'p = {p}, f = {f}')

dfdp = 0
dfdz = 1

dpdx = y
dpdy = x

dfdx = dfdp * dpdx
dfdy = dfdp * dpdy
print(f'dfdx : {dfdx}, dfdy : {dfdy}, dfdz = {dfdz}')
```

```
p = -10, f = -4
dfdx : 0, dfdy : 0, dfdz = 1
```

4번

4-1

In [5]:

```
def sigmoid(x):
    return 1/(1+np.exp(-x))
```

In [6]:

```
input_ = np.array([0.4,0.5])
input_w = np.array([[2.0,3.0],[1,4]])
output = np.dot(input_,input_w)
final = sigmoid(output)
print(final)
```

```
[0.78583498 0.96083428]
```

4-2

In [7]:

```
target=np.array([1.70887704,1.4168273]) # 목표값
error = abs(target-output) # error의 절대값
w = input_w.copy()
normalize = np.array([[w[0,0]/(w[0,0]+w[1,0]),w[0,1]/(w[0,1]+w[1,1])],
                 [w[1,0]/(w[1,0]+w[0,0]),w[1,1]/(w[0,1]+w[1,1])]]) # 정규화
hidden_error = np.dot(normalize,error)
print(hidden_error)
```

```
[1.03680156 1.15524818]
```

4-3

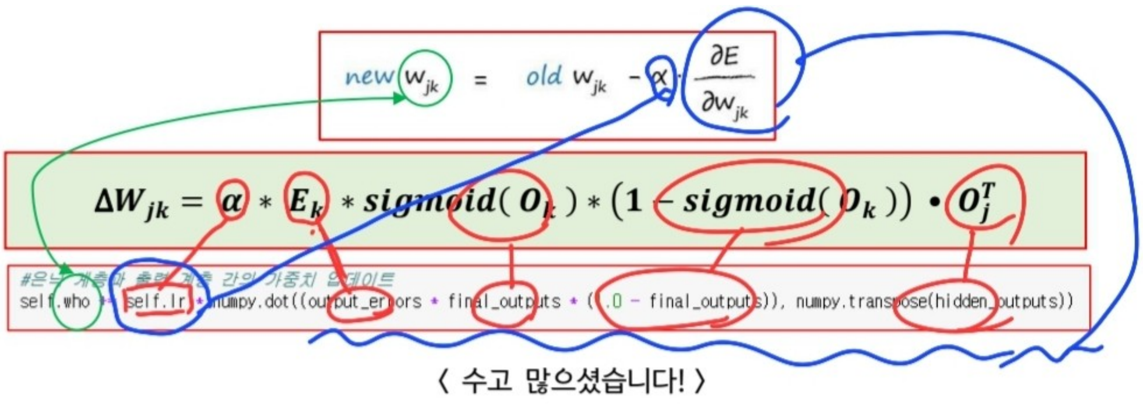
In [8]:

```
learning_rate=0.1
input_w -= learning_rate*np.dot(-error* output* (1-output),input_w.T)
input_w
```

Out[8]:

```
array([[-1.79795315, -2.03736053],
       [-2.79795315, -1.03736053]])
```

5번



$$\text{new } w_{jk} = \text{old } w_{jk} - \alpha \cdot \frac{\partial E}{\partial w_{jk}}$$

$$\Delta W_{jk} = \alpha * E_k * sigmoid(O_k) * (1 - sigmoid(O_k)) \bullet O_j^T$$

```
#은닉 계층과 출력 계층 간의 가중치 업데이트
self.who += self.lr * numpy.dot((output_errors * final_outputs * (1.0 - final_outputs)), numpy.transpose(hidden_outputs))
```

〈 수고 많으셨습니다! 〉