EXCEL로 만든 인공신경망을 pytorch의 Tensor 클래스을 이용하여 만들기

In [1]:
```
import torch
```

필요한 함수의 정의하기

In [2]:
```
def ReLU_func(outputs):
    # 채우기

    #모듈 사용
    #m = torch.nn.ReLU()
    #final_outputs = m(outputs)

    #직접구현
    final_outputs = torch.clamp(outputs,0)
    return final_outputs

def softmax(outputs):
    # 채우기
    #모듈 사용
    m = torch.nn.Softmax(dim=1)
    softmax = m(outputs)

    #직접구현
#     softmax = torch.Tensor()
#     for output in outputs:
#         softmax = torch.cat((softmax, torch.exp(torch.unsqueeze(output, 0)) / torch.sum(torch.exp(torch.unsquee
    return softmax

def cross_entropy(outputs, labels):
    return torch.sum(-labels * torch.log(outputs),1)
```

Weight 값 설정하기

In [3]:
```
w_ih = torch.Tensor([[1,-2,3],
                     [-2,5,3]])
w_ho = torch.Tensor([[3,-2,4],
                     [-1,2,3],
                     [2,-2,-4]])
```

Batch 가 3인 Input 값과 Labels 값의 입력하기

In [4]:
```
input = torch.Tensor([[2,-4],
                      [0,3],
                      [-6,2]])
labels = torch.Tensor([[1,0,0],
                      [0,1,0],
                      [0,0,1]])
```

L1의 활성화 함수 전의 값 구하기

In [5]:
```
L1 = input.matmul(w_ih)
print(L1)
```

```
tensor([[ 10., -24.,  -6.],
        [ -6.,  15.,   9.],
        [-10.,  22., -12.]])
```

ReLU의 적용하기

In [6]:
```
L1 = ReLU_func(L1)
print(L1)
```

```
tensor([[10.,  0.,  0.],
        [ 0., 15.,  9.],
        [ 0., 22.,  0.]])
```

Output 구하기

In [7]:
```
outputs = L1.matmul(w_ho)
print(outputs)
```

```
tensor([[ 30., -20.,  40.],
        [  3.,  12.,   9.],
        [-22.,  44.,  66.]])
```

Softmax의 적용

In [8]:
```
so = softmax(outputs)
print(so)
```

```
tensor([[4.5398e-05, 8.7561e-27, 9.9995e-01],
        [1.1754e-04, 9.5246e-01, 4.7420e-02],
        [6.0546e-39, 2.7895e-10, 1.0000e+00]])
```

sample별로 loss 구하기

In [9]:
```
loss = cross_entropy(so,labels)
print(loss)
```

```
tensor([10.0000,  0.0487,  0.0000])
```

최종 loss 구하기

In [10]:
```
loss = torch.mean(loss)
print(loss)
```

```
tensor(3.3496)
```

EXCEL과 같은 답이 나오는가?

아니면 이유가 무엇인가?