In [23]:
```
import torch
import math
import numpy as np
```

- Hoàn thành công thức tính loss function.

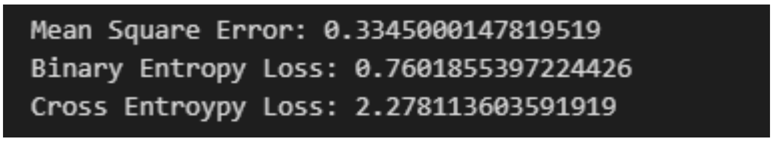
In [24]:
```
# Công thức tính CrossEntropy Loss
def crossEntropyLoss(output, target):
  if len(output) != len(target):
    return 0
  n = len(output)
  return sum([-target[x]*math.log(output[x]) for x in range(n)])
```

In [25]:
```
# Công thức tính Mean Square Error
def meanSquareError(output, target):
  if len(output) != len(target):
    return 0
  n = len(output)
  return sum([(target[x] - output[x])**2 for x in range(n)])/n
```

In [26]:
```
# Công thức tính BinaryEntropy Loss
def binaryEntropyLoss(output, target, n):
  bce = lambda y, p: y * math.log(p) + (1 - y) * math.log(1 - p)
  return -sum([bce(target[i], output[i]) for i in range(n)])/n
```

In [27]:
```
inputs = torch.tensor([0.1, 0.3, 0.6, 0.7])
target = torch.tensor([0.31, 0.32, 0.8, 0.2])
n = len(inputs)
mse = meanSquareError(inputs, target)
binary_loss = binaryEntropyLoss(inputs, target, n)
cross_loss = crossEntropyLoss(inputs, target)
print(f"Mean Square Error: {mse}")
print(f"Binary Entropy Loss: {binary_loss}")
print(f"Cross Entroypy Loss: {cross_loss}")
```

```
Mean Square Error: 0.08362500369548798
Binary Entropy Loss: 0.7601855397224426
Cross Entroypy Loss: 1.5790680646896362
```



```
Mean Square Error: 0.3345000147819519
Binary Entropy Loss: 0.7601855397224426
Cross Entroypy Loss: 2.278113603591919
```

- Hoàn thành công thức tính activation function

In [28]:
```
# Công thức hàm sigmoid
def sigmoid(x: torch.tensor):
  return 1 / (1 + np.exp(-x))
```

In [29]:
```
# Công thức hàm relu
def relu(x: torch.tensor):
  return np.maximum(0, x)
```

In [46]:

```
# Công thức hàm softmax
def softmax(zi: torch.tensor):
  e_Z = np.exp(zi)
  return e_Z / e_Z.sum(axis = 0)
```

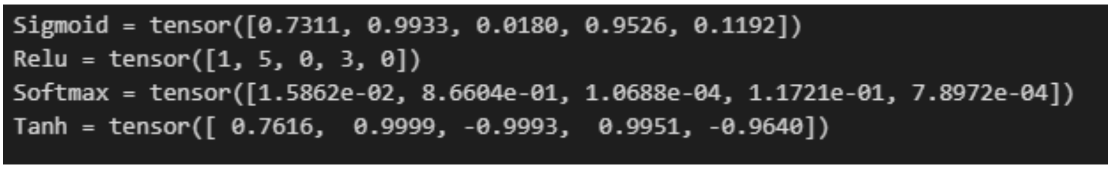
In [38]:

```
# Công thức hàm tanh
def tanh(x: torch.tensor):
  return np.tanh(x)
```

In [47]:

```
x = torch.tensor([1, 5, -4, 3, -2])
f_sigmoid = sigmoid(x)
f_relu = relu(x)
f_softmax = softmax(x)
f_tanh = tanh(x)
print(f"Sigmoid = {f_sigmoid}")
print(f"Relu = {f_relu}")
print(f"Softmax = {f_softmax}")
print(f"Tanh = {f_tanh}")
```

```
Sigmoid = tensor([0.7311, 0.9933, 0.0180, 0.9526, 0.1192], dtype=torch.flo
at64)
Relu = tensor([1, 5, 0, 3, 0])
Softmax = tensor([1.5862e-02, 8.6604e-01, 1.0688e-04, 1.1721e-01, 7.8972
e-04],
       dtype=torch.float64)
Tanh = tensor([ 0.7616,  0.9999, -0.9993,  0.9951, -0.9640], dtype=torch.f
loat64)
```



```
Sigmoid = tensor([0.7311, 0.9933, 0.0180, 0.9526, 0.1192])
Relu = tensor([1, 5, 0, 3, 0])
Softmax = tensor([1.5862e-02, 8.6604e-01, 1.0688e-04, 1.1721e-01, 7.8972e-04])
Tanh = tensor([ 0.7616,  0.9999, -0.9993,  0.9951, -0.9640])
```