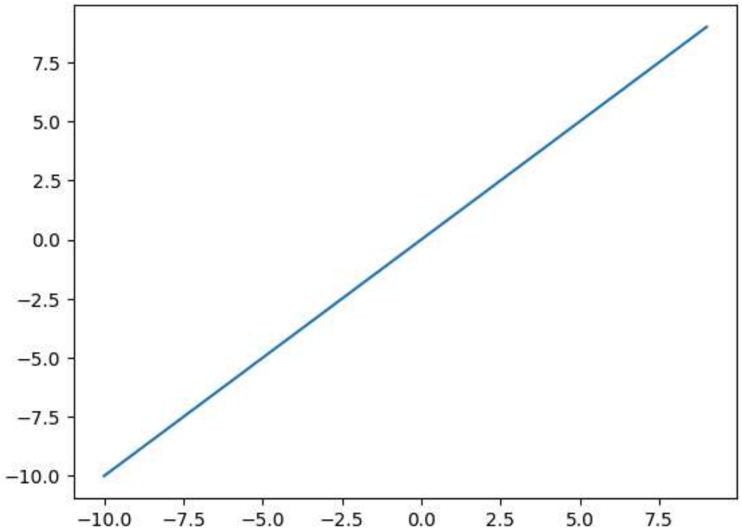
```
import matplotlib.pyplot as plt
```

```
x = [i for i in range (-10, 10)]
```

```
plt.plot (x, x)
```

```
[<matplotlib.lines.Line2D at 0x7f75a82443d0>]
```



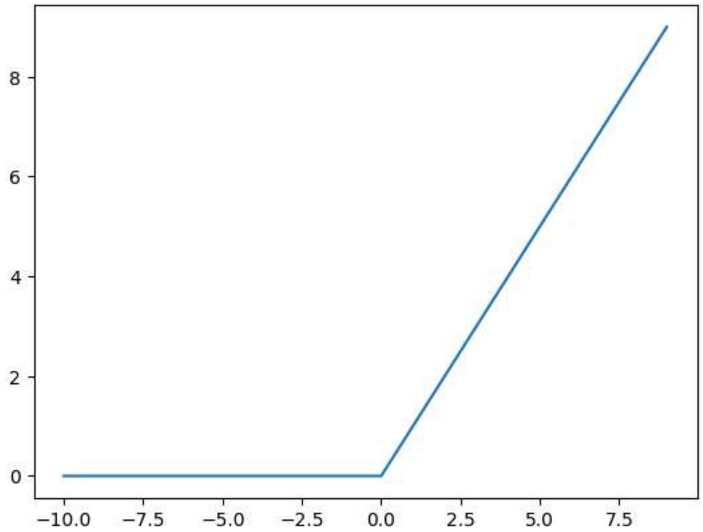
```
def rectifiedLinearUnit (x):
  return max (0.0, x)
```

```
def rectifiedLinearUnit (x):
  return max (0.0, x)
```

```
output = [rectifiedLinearUnit (i) for i in x]
```

```
plt.plot (x, output)
```

```
[<matplotlib.lines.Line2D at 0x7f75a82861d0>]
```



```
import torch
from torch.nn import ReLU

tensor_X = torch.range (start=-10, end=10, step=1)
relu_object = ReLU ()
```

```
torch_output = relu_object (tensor_X)
torch_output
```

```
/tmp/ipython-input-9-2188386771.py:4: UserWarning: torch.range is deprecated and will be removed in a future release because its behavic
  tensor_X = torch.range (start=-10, end=10, step=1)
tensor([ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.,  2.,  3.,
         4.,  5.,  6.,  7.,  8.,  9., 10.])
```

```
import tensorflow as tf
from tensorflow.nn import relu

tf_tensor = tf.range (start=-10, limit=10)
tf_output = relu (tf_tensor).numpy ()
tf_output
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9],
      dtype=int32)
```

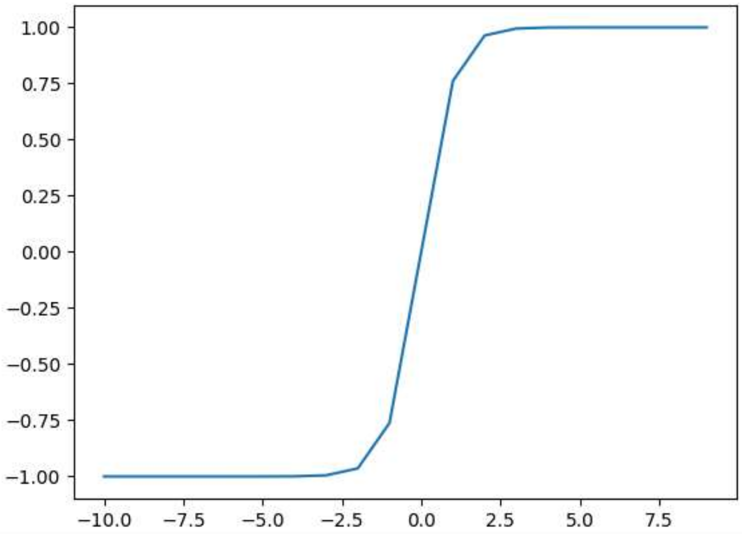
```
from math import exp
x = [i for i in range (-10 ,10)]

def hyperbolicTanget (x):
  return (exp (x) - exp (-x)) / (exp (x) + exp (-x))

output = [hyperbolicTanget (i) for i in x]
```

```
plt.plot (x, output)
```

```
[<matplotlib.lines.Line2D at 0x7f743929acd0>]
```



```
from torch.nn import Tanh

tensor_X = torch.range (start=-10, end=10, step=1)
relu_object = Tanh ()
torch_output = relu_object (tensor_X)
torch_output
```

```
/tmp/ipython-input-13-1813261820.py:3: UserWarning: torch.range is deprecated and will be removed in a future release because its behavi
  tensor_X = torch.range (start=-10, end=10, step=1)
tensor([-1.0000, -1.0000, -1.0000, -1.0000, -1.0000, -0.9999, -0.9993, -0.9951,
        -0.9640, -0.7616,  0.0000,  0.7616,  0.9640,  0.9951,  0.9993,  0.9999,
         1.0000,  1.0000,  1.0000,  1.0000,  1.0000])
```

```
from tensorflow.nn import tanh

tf_tensor = tf.range (start=-10.0, limit=10.0)
tf_output = tanh (tf_tensor).numpy ()
tf_output
```

```
array([-1.        , -1.        , -1.        , -0.99999833, -0.99998784,
       -0.99990916, -0.9993292 , -0.9950547 , -0.9640276 , -0.7615942 ,
        0.        ,  0.7615942 ,  0.9640276 ,  0.9950547 ,  0.9993292 ,
        0.99990916,  0.99998784,  0.99999833,  1.        ,  1.        ],
      dtype=float32)
```

```
import numpy as np
np.random.seed (7)

output_layer_vector = np.random.randn (1, 9)
print (output_layer_vector)
```

```
[[ 1.69052570e+00 -4.65937371e-01  3.28201637e-02  4.07516283e-01
  -7.88923029e-01  2.06557291e-03 -8.90385858e-04 -1.75472431e+00
   1.01765801e+00]]
```

```
def softmax (x):
  return np.exp (x) / np.exp (x).sum ()
```

```
result = softmax (output_layer_vector)
result
```

```
array([[0.38782192, 0.04488408, 0.07390945, 0.10750494, 0.03249537,
        0.07167099, 0.07145945, 0.01237027, 0.19788352]])
```

```
np.argmax (result)
```

```
np.int64(0)
```

```
from torch.nn import Softmax

torch.manual_seed (7)
x = torch.randn ([1, 5])

softmax = Softmax ()
output = softmax (x)

print (f'Output Vector : {x}')
print (f'Softmax Vector : {output}')
```

```
Output Vector : tensor([[-0.1468,  0.7861,  0.9468, -1.1143,  1.6908]])
Softmax Vector : tensor([[0.0758, 0.1927, 0.2263, 0.0288, 0.4763]])
/usr/local/lib/python3.11/dist-packages/torch/nn/modules/module.py:1739: UserWarning: Implicit dimension choice for softmax has been dep
  return self._call_impl(*args, **kwargs)
```

```
from tensorflow.nn import softmax

tf.random.set_seed (7)
x = tf.random.normal ([1, 5], dtype = tf.float64)
output = softmax (x)
print (f'Output Vector : {x}')
print (f'Softmax Vector : {output}')
```

```
Output Vector : [[ 0.20519253  0.86145926 -0.86256865  0.97639899  0.6010404 ]]
Softmax Vector : [[0.14452169 0.2785775  0.04968328 0.31250986 0.21470767]]
```