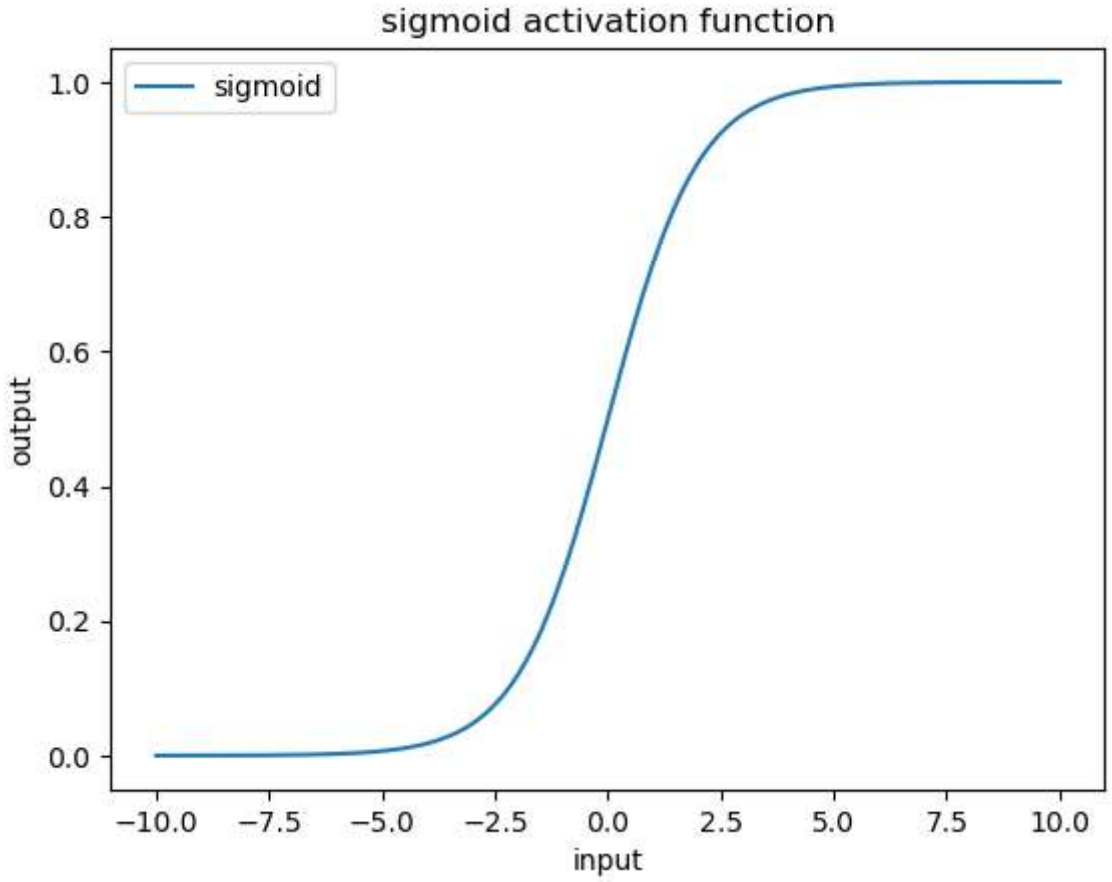
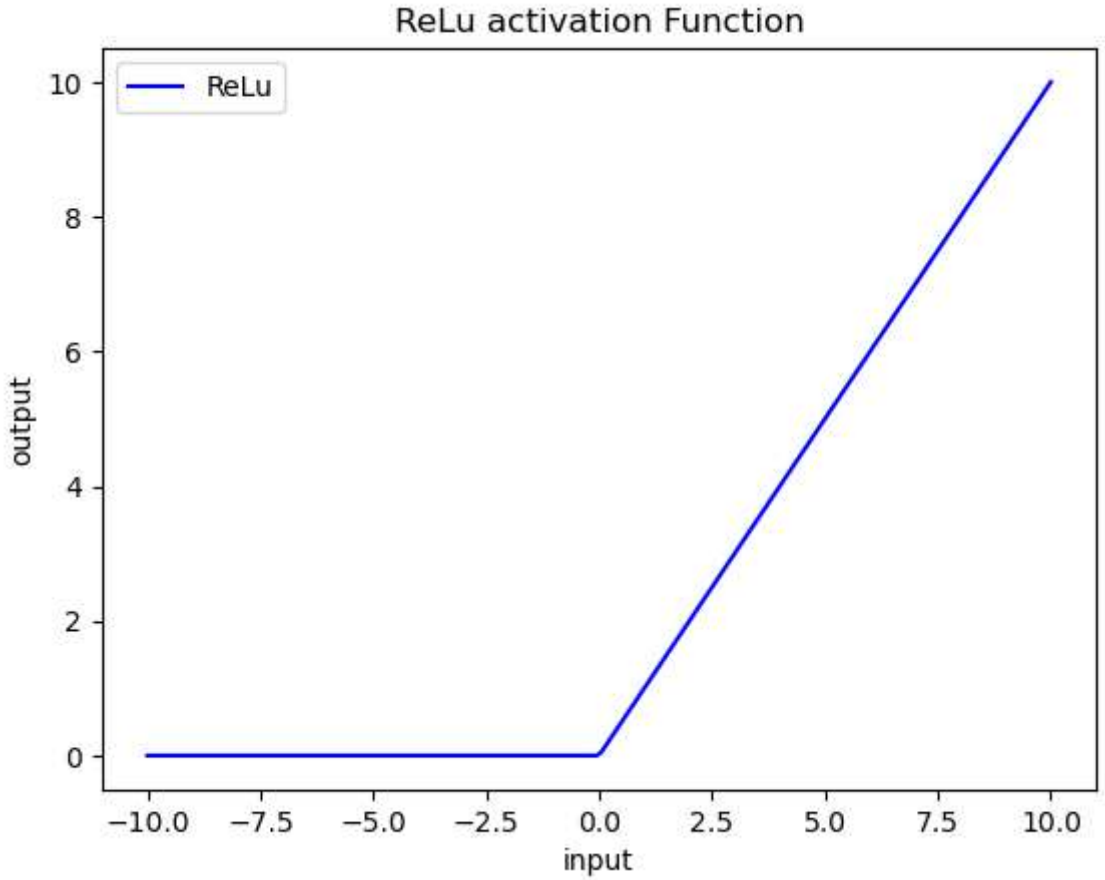
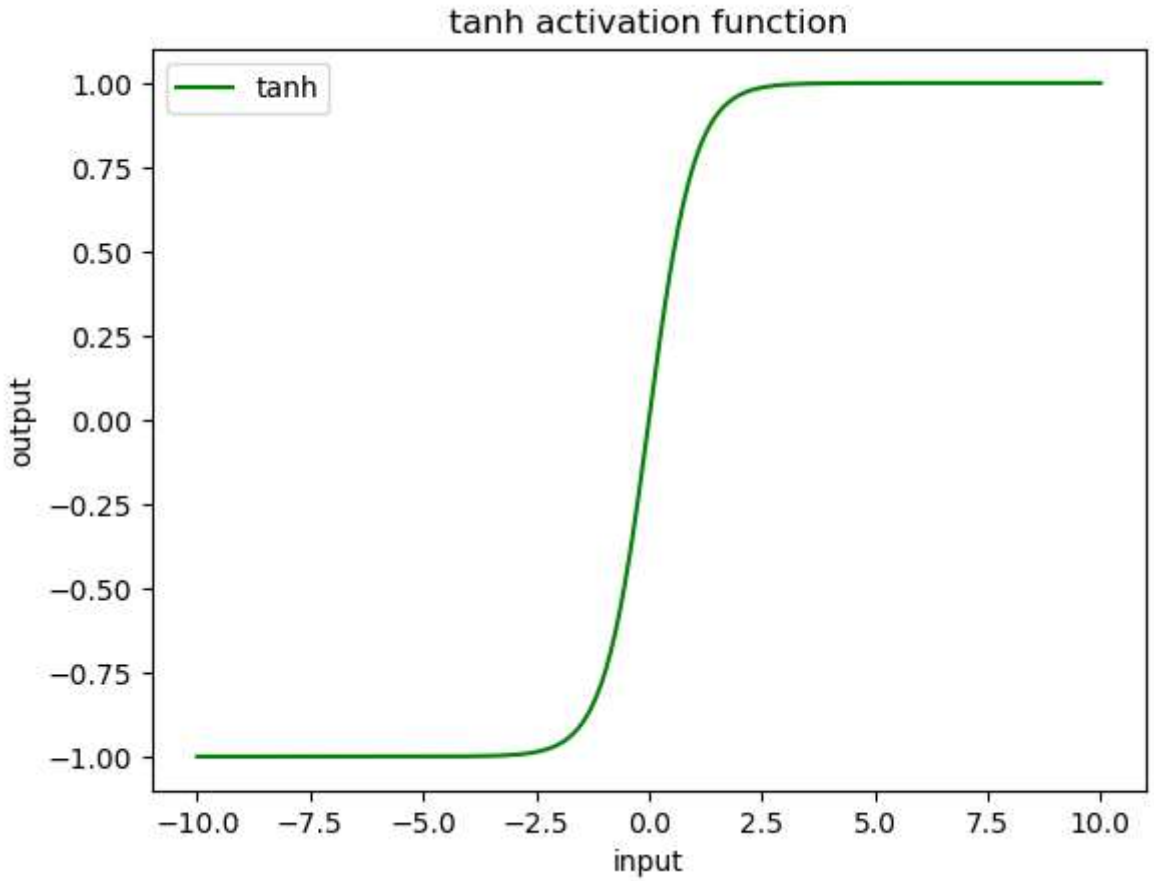
In [1]:
```
#SIGMOID FUNCTION
import numpy as np
def sig(x):
    return 1/(1+np.exp(-x))
x=np.linspace(-10,10,200)
y=sig(x)
import matplotlib.pyplot as plt
plt.plot(x,y,label='sigmoid')
plt.title('sigmoid activation function')
plt.xlabel('input')
plt.ylabel('output')
plt.legend()
plt.show()
```



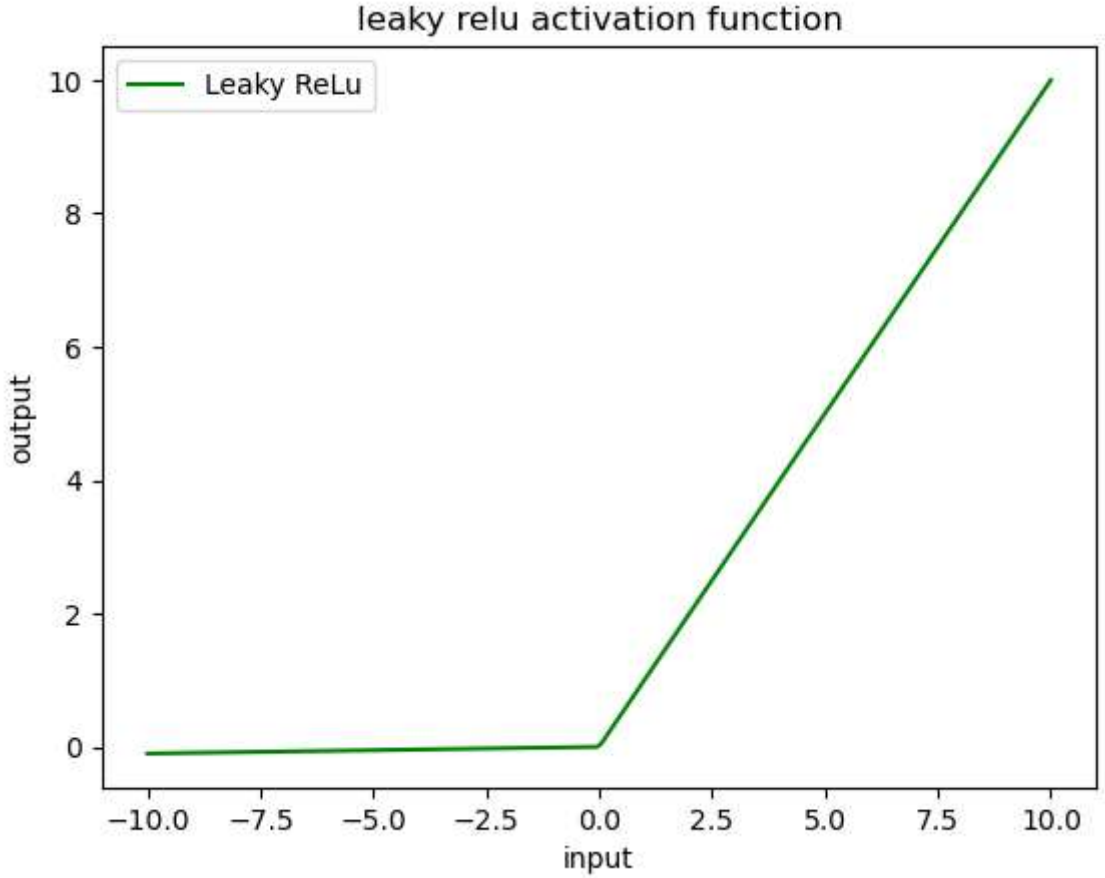
In [2]:
```
#RELU
def relu(x):
    return np.maximum(0,x)
y=relu(x)
plt.plot(x,y,label='ReLu',color='blue')
plt.title('ReLu activation Function')
plt.xlabel('input')
plt.ylabel('output')
plt.legend()
plt.show()
```



In [3]:
```
#TAN FUNction
def tanh(x):
    return np.tanh(x)
y=tanh(x)
plt.plot(x,y,label='tanh',color='green')
plt.title('tanh activation function')
plt.xlabel('input')
plt.ylabel('output')
plt.legend()
plt.show()
```



In [4]:
```
#LEaky RELU
def leaky_relu(x,alpha=0.01):
    return np.where(x>0,x,alpha*x)
y=leaky_relu(x)
plt.plot(x,y,label='Leaky ReLu',color='green')
plt.title('leaky relu activation function')
plt.xlabel('input')
plt.ylabel('output')
plt.legend()
plt.show()
```



In [9]:
```
#softmax
def softmax(x):
    exp_x=np.exp(x-np.max(x,axis=-1,keepdims=True))
    return exp_x/np.sum(exp_x,axis=-1,keepdims=True)
scores=np.array([4.0,2.0,0.3])
probs=softmax(scores)
print('softmax probabilities:',probs)
```

```
softmax probabilities: [0.86202526 0.11666243 0.0213123 ]
```

In [ ]: