Name:KALYANAM VENKATA SREE SAI

ID:2000030439

SEC:01

CourseCode:20cs3026RA

In [2]:

```
import numpy as np
import matplotlib.pyplot as plt
```

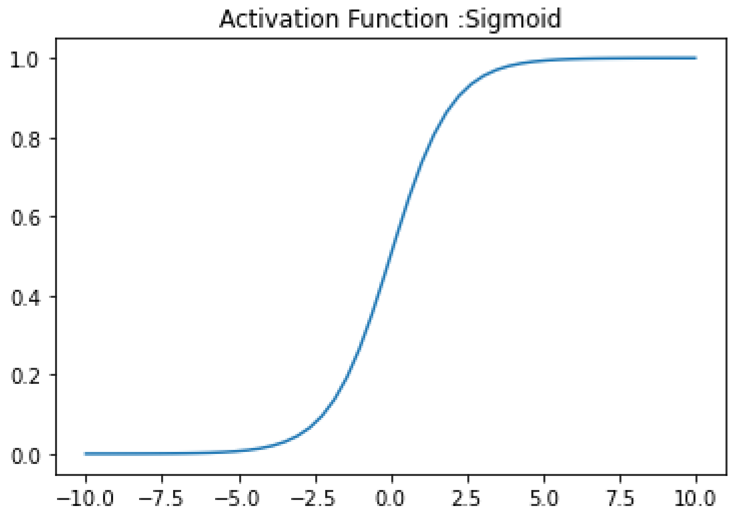
sigmoid activation function

In [3]:

```
def sigmoid(x):


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

In [4]:

```
x = np.linspace(-10, 10)
plt.plot(x, sigmoid(x))
plt.axis('tight')
plt.title('Activation Function :Sigmoid')
plt.show()
```



Tanh activation function

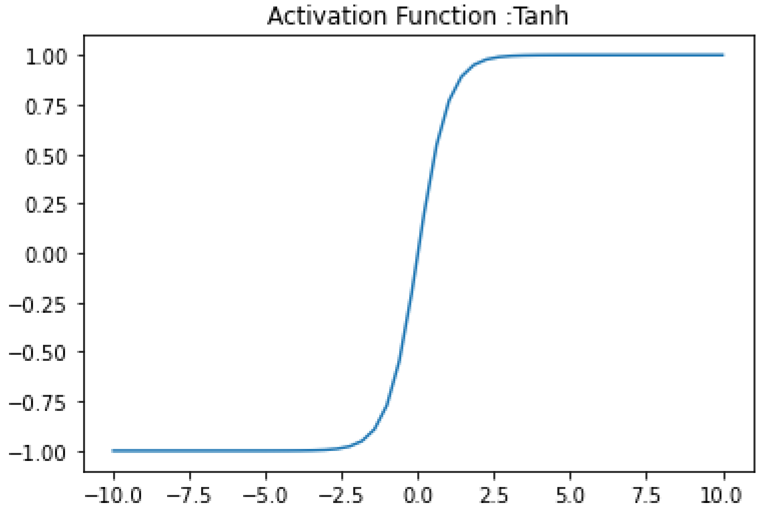
In [5]:

```
def tanh(x):


    return np.tanh(x)
```

In [6]:

```
x = np.linspace(-10, 10)
```

```
plt.plot(x, tanh(x))
plt.axis('tight')
plt.title('Activation Function :Tanh')
plt.show()
```



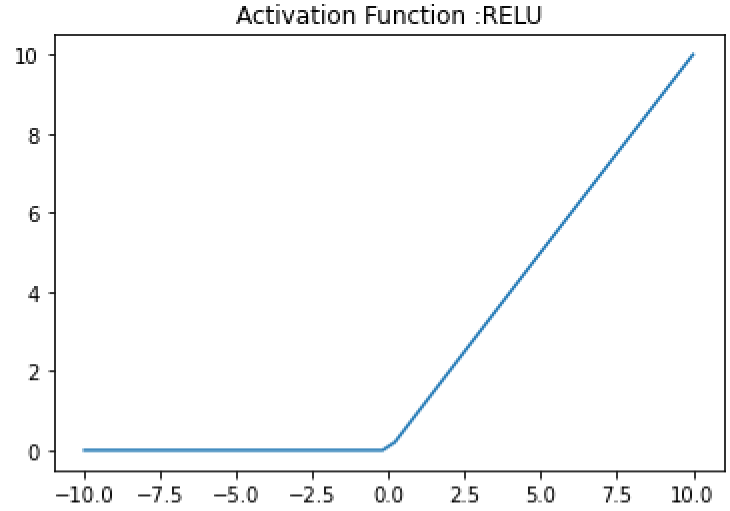
RELU Activation Function

In [7]:

```
def RELU(x):
    x1=[]
    for i in x:
        if i<0:
            x1.append(0)
        else:
            x1.append(i)

    return x1
```

In [8]:

```
x = np.linspace(-10, 10)
plt.plot(x, RELU(x))
plt.axis('tight')
plt.title('Activation Function :RELU')
plt.show()
```



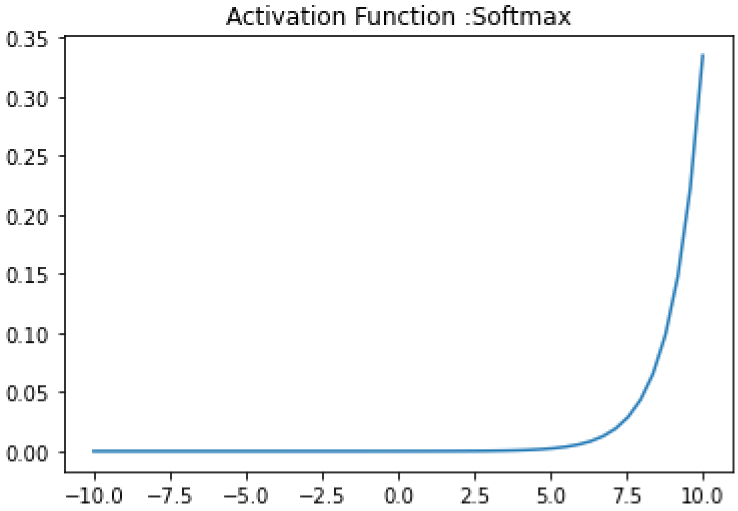
Softmax Activation Function

In [9]:

```
def softmax(x):
    return np.exp(x) / np.sum(np.exp(x), axis=0)
```

In [10]:

```
x = np.linspace(-10, 10)
plt.plot(x, softmax(x))
plt.axis('tight')
plt.title('Activation Function :Softmax')
plt.show()
```



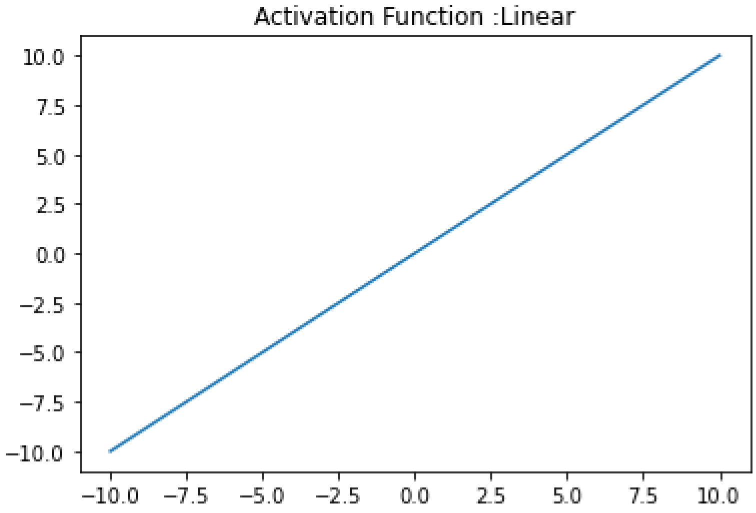
Linear activation function

In [11]:

```
def linear(x):
    return x
```

In [12]:

```
x = np.linspace(-10, 10)
plt.plot(x, linear(x))
plt.axis('tight')
plt.title('Activation Function :Linear')
plt.show()
```

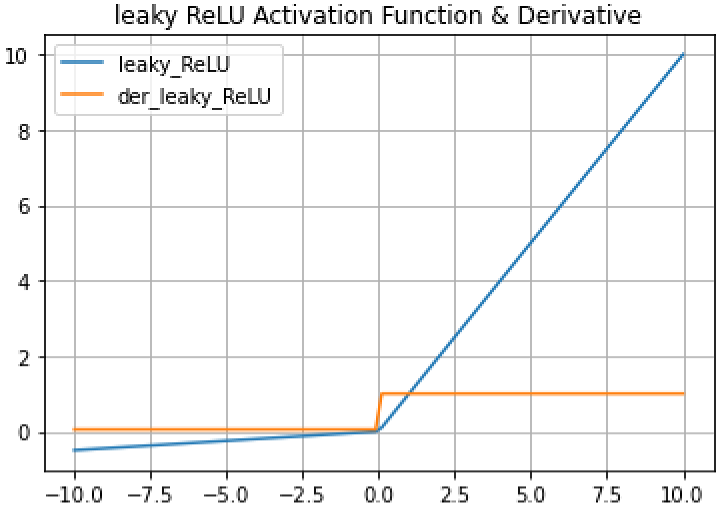


Leaky RELU activation function

In [13]:

```
import numpy as np
import matplotlib.pyplot as plt

# Leaky Rectified Linear Unit (leaky ReLU) Activation Function
```

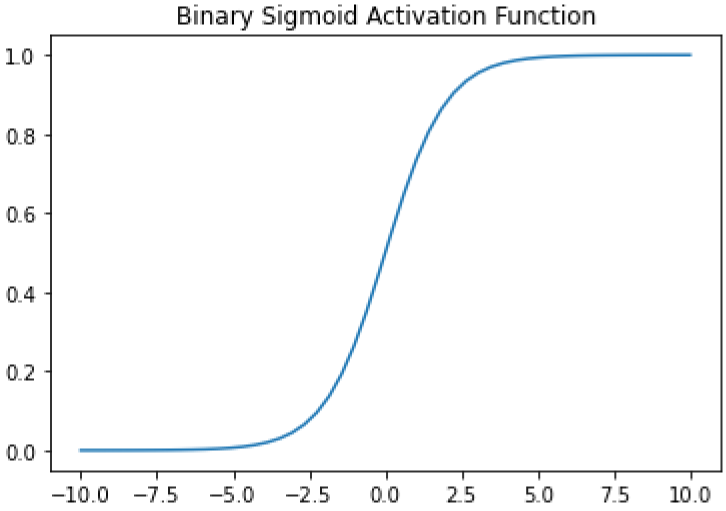
```
def leaky_ReLU(x):
  data = [max(0.05*value,value) for value in x]
  return np.array(data, dtype=float)

# Derivative for leaky ReLU
def der_leaky_ReLU(x):
  data = [1 if value>0 else 0.05 for value in x]
  return np.array(data, dtype=float)

# Generating data For Graph
x_data = np.linspace(-10,10,100)
y_data = leaky_ReLU(x_data)
dy_data = der_leaky_ReLU(x_data)

# Graph
plt.plot(x_data, y_data, x_data, dy_data)
plt.title('leaky ReLU Activation Function & Derivative')
plt.legend(['leaky_ReLU','der_leaky_ReLU'])
plt.grid()
plt.show()
```



binary sigmoid

In [14]:

```
import numpy as np
import matplotlib.pyplot as plt
import numpy as np

def SigmoidBinary(t):
    return 1/(1+np.exp(-t))
t = np.linspace(-10, 10)
plt.plot(t, SigmoidBinary(t))
plt.axis('tight')
plt.title('Binary Sigmoid Activation Function')
plt.show()
```


Binary Sigmoid Activation Function
1.0
0.8
0.6
0.4
0.2
0.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0