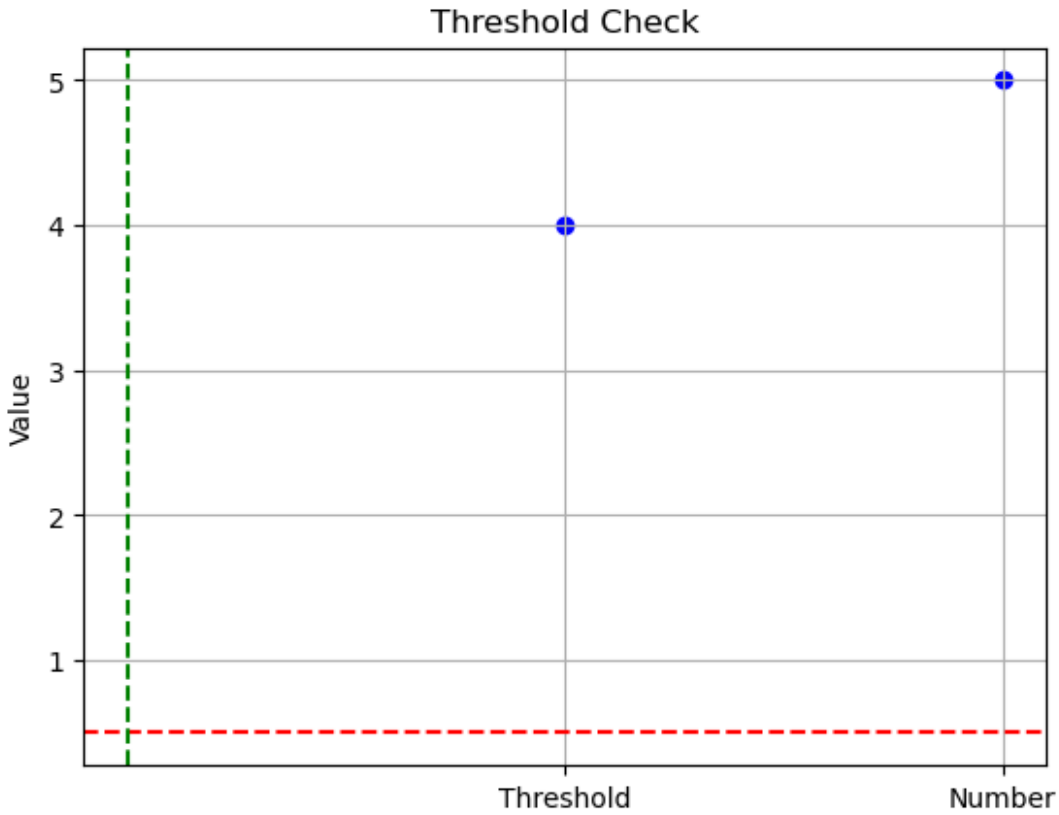
```python
#1
#Threshold Function
import matplotlib.pyplot as plt
vk=int(input("enter threshold value:"))
n=int(input("enter number:"))
if n>vk:
  print("activated - 1")
else:
  print("not activated - 0")
xval= [1, 2]
yval= [vk, n]
plt.scatter(xval, yval, color='blue')
plt.xticks(xval, ['Threshold', 'Number'])
plt.axhline(0.5, color='red', linestyle='--')
plt.axvline(0, color='green', linestyle='--')
plt.ylabel('Value')
plt.title('Threshold Check')
plt.grid()
plt.show()
```

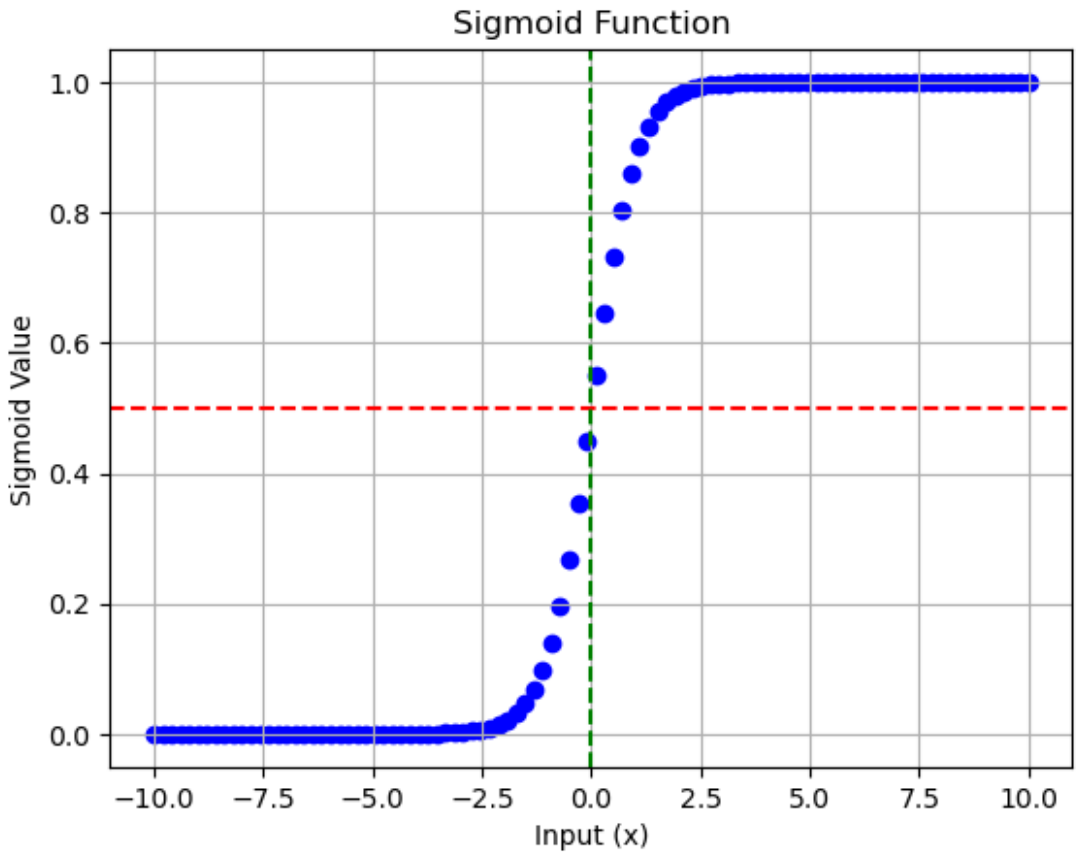
```
enter threshold value: 4
enter number: 5

activated - 1
```



```
#Sigmoid Function
import numpy as np
x=int(input("enter slope:"))
y=1/(1+np.exp(-x))
print(y)
xval= np.linspace(-10, 10, 100)
yval= 1 / (1 + np.exp(-xval * x))
plt.scatter(xval, yval, color='blue')
plt.axhline(0.5, color='red', linestyle='--')
plt.axvline(0, color='green', linestyle='--')
plt.title('Sigmoid Function')
plt.xlabel('Input (x)')
plt.ylabel('Sigmoid Value')
plt.grid()
plt.show()
```

```
enter slope: 2
```

```
0.8807970779778823
```

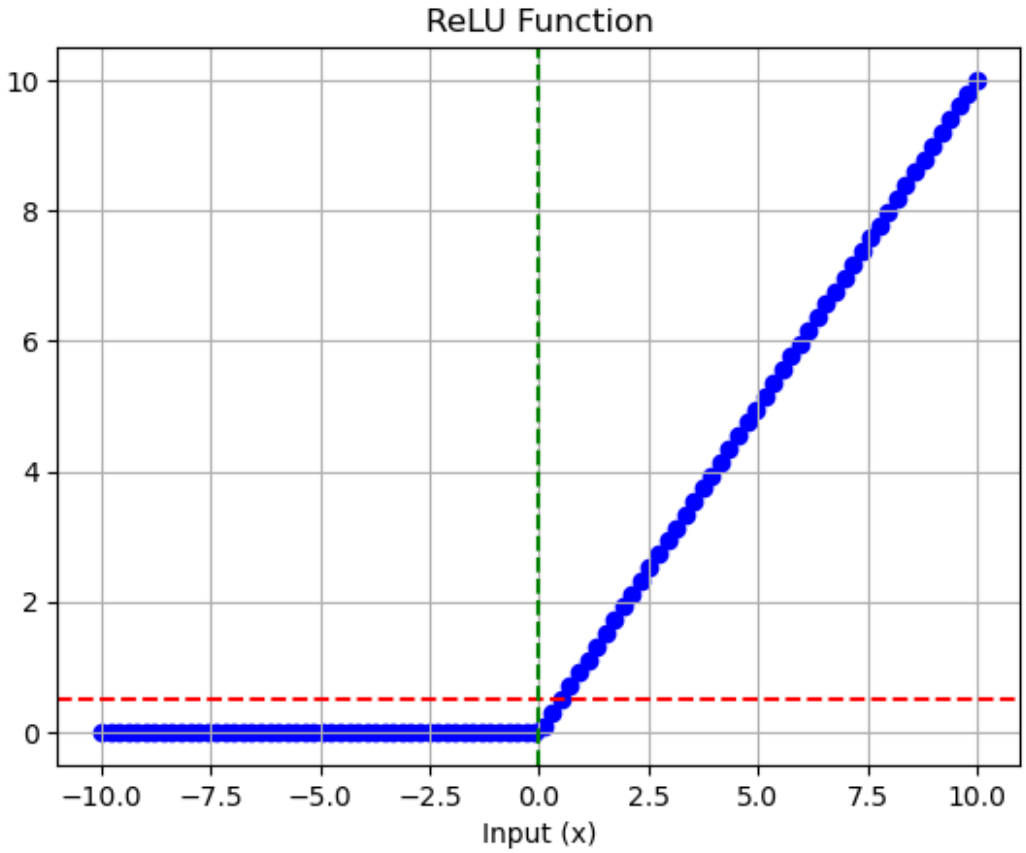



```
#ReLU
x=int(input("enter number:"))
fx=max(0,x)
```

```
print(fx)
xval= np.linspace(-10, 10, 100)
yval= np.maximum(0,xval)
plt.scatter(xval, yval, color='blue')
plt.axhline(0.5, color='red', linestyle='--')
plt.axvline(0, color='green', linestyle='--')
plt.title('ReLU Function')
plt.xlabel('Input (x)')
plt.grid()
plt.show()
```

```
enter number: 3
```

```
3
```

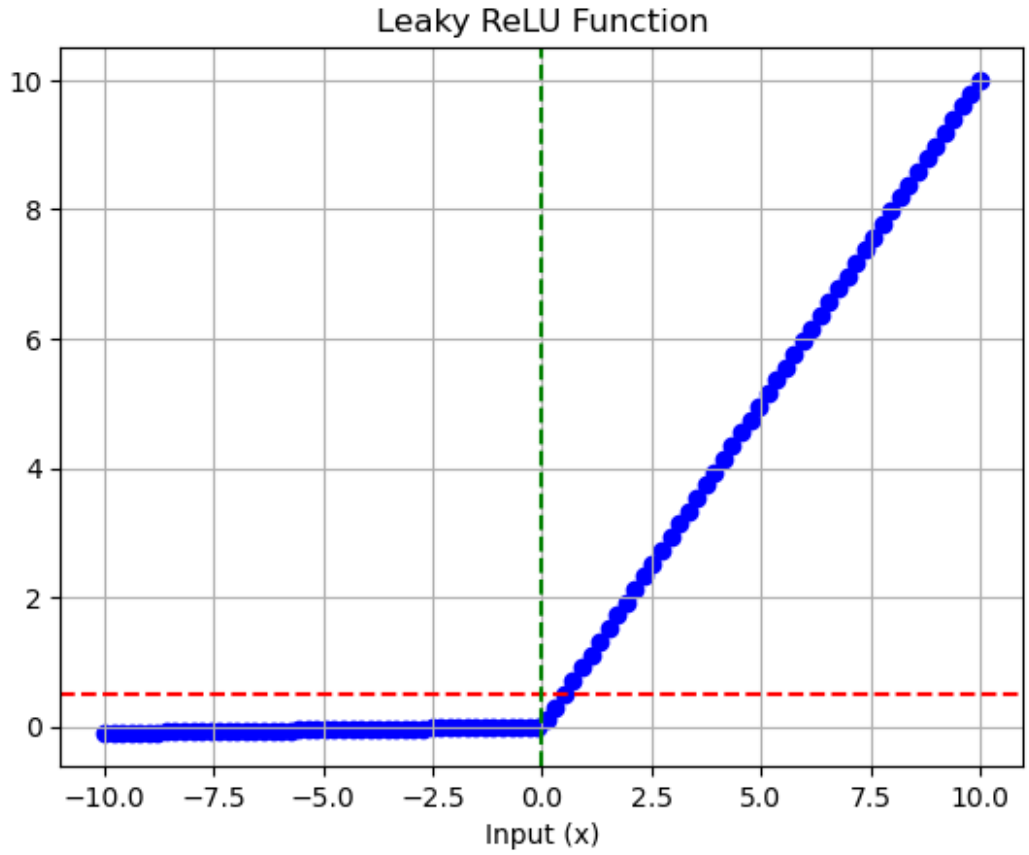



```
#Leaky ReLU Fucntion
x=int(input("enter number:"))
fx=0.01*x
print(fx)
xval= np.linspace(-10, 10, 100)
yval= np.where(xval > 0, xval, 0.01 * xval)
plt.scatter(xval, yval, color='blue')
plt.axhline(0.5, color='red', linestyle='--')
```

```
plt.axvline(0, color='green', linestyle='--')
plt.title('Leaky ReLU Function')
plt.xlabel('Input (x)')
plt.grid()
plt.show()
```

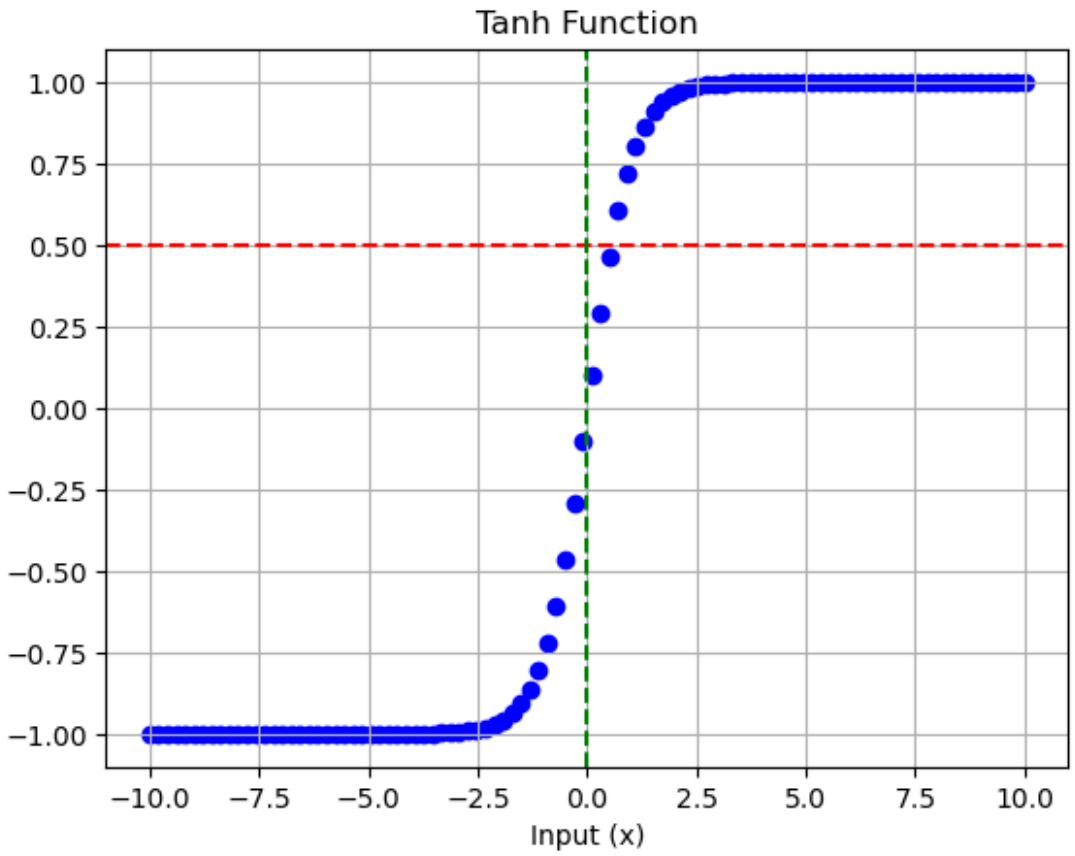
```
enter number: 3
```

```
0.03
```



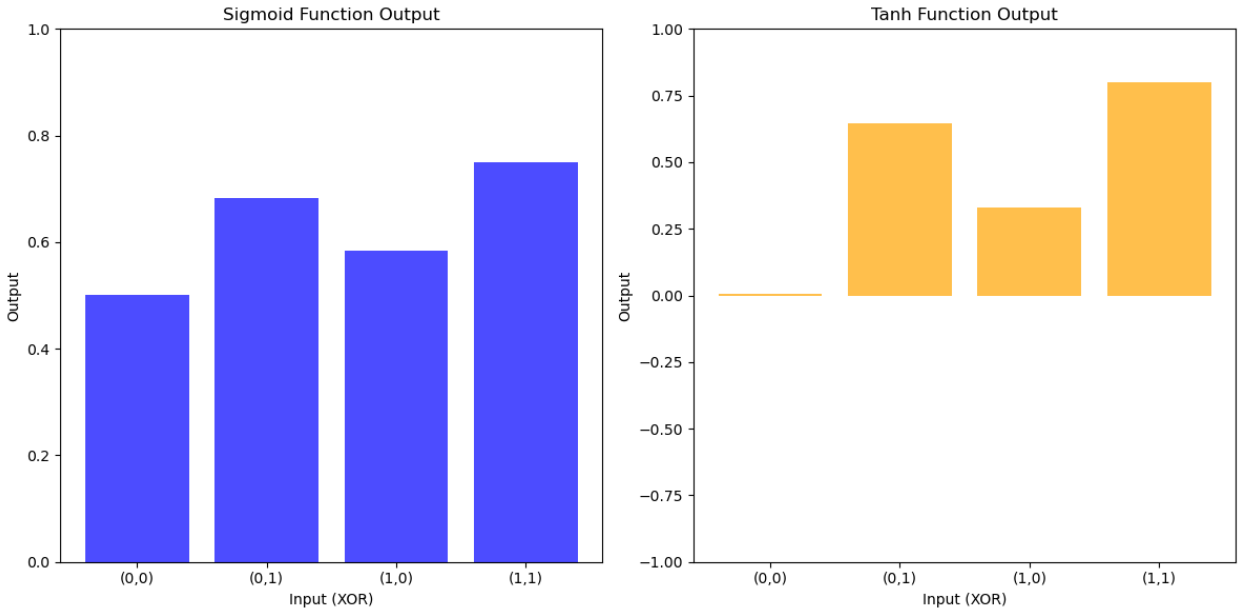

```
#Tanh Function
import numpy as np
x=int(input("enter number:"))
tanh=(np.exp(x)-np.exp(-x))/(np.exp(x)+np.exp(-x))
print(tanh)
xval= np.linspace(-10, 10, 100)
yval= (np.exp(xval)-np.exp(-xval))/(np.exp(xval)+np.exp(-xval))
plt.scatter(xval, yval, color='blue')
plt.axhline(0.5, color='red', linestyle='--')
plt.axvline(0, color='green', linestyle='--')
plt.title('Tanh Function')
plt.xlabel('Input (x)')
```

```
plt.grid()
plt.show()

enter number: 2

0.964027580075817
```




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

def sigmoid(x):
    return 1 / (1 + np.exp(-x))

def tanh(x):
    return (np.exp(x) - np.exp(-x)) / (np.exp(x) + np.exp(-x))

# Random weights and bias
weights = np.random.rand(2)
bias = np.random.rand(1)

# XOR input values
xor = np.array([[0, 0], [0, 1], [1, 0], [1, 1]])
output1 = []
```

```
output2 = []

# Calculate outputs for sigmoid and tanh
for i in xor:
    sum_val = np.dot(weights, i) + bias
    output1.append(sigmoid(sum_val))
    output2.append(tanh(sum_val))

# Convert outputs to NumPy arrays
output1 = np.array(output1).flatten()
output2 = np.array(output2).flatten()

# Print the outputs
print("sigmoid:", output1)
print("tanh:", output2)
plt.figure(figsize=(12, 6))
# Sigmoid plot
plt.subplot(1, 2, 1)
plt.title("Sigmoid Function Output")
plt.bar(range(len(xor)), output1, color='blue', alpha=0.7)
plt.xticks(range(len(xor)), ['(0,0)', '(0,1)', '(1,0)', '(1,1)'])
plt.ylim(0, 1)
plt.ylabel("Output")
plt.xlabel("Input (XOR)")
# Tanh plot
plt.subplot(1, 2, 2)
plt.title("Tanh Function Output")
plt.bar(range(len(xor)), output2, color='orange', alpha=0.7)
plt.xticks(range(len(xor)), ['(0,0)', '(0,1)', '(1,0)', '(1,1)'])
plt.ylim(-1, 1)
plt.ylabel("Output")
plt.xlabel("Input (XOR)")
plt.tight_layout()
plt.show()

sigmoid: [0.50134752 0.68245184 0.58443845 0.75038995]
tanh: [0.00539005 0.64404919 0.32838837 0.80074786]
```


Sigmoid Function Output
Output
Input (XOR)
(0,0)
(0,1)
(1,0)
(1,1)
0.0
0.2
0.4
0.6
0.8
1.0
Tanh Function Output
Output
Input (XOR)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00