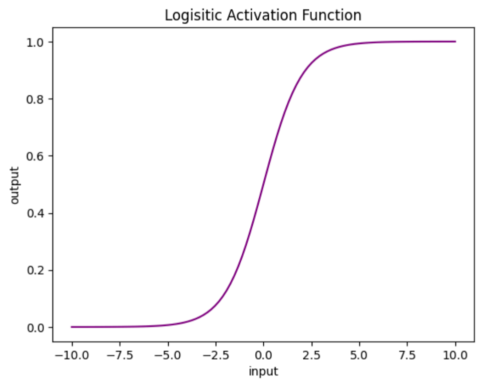
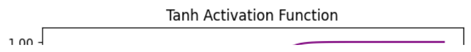
```
import torch
import torch.nn as nn
import torch.nn.functional as F


class SimpleNN(nn.Module):
  def __init__(self):

    super(SimpleNN,self).__init__()

    self.fc1 = nn.Linear(784,128)
    self.fc2 = nn.Linear(128,10)
    self.fc3 = nn.Linear(10,1)

  def forward(self,x):
    x = F.relu(self.fc1(x))
    x = F.relu(self.fc2(x))
    x = F.log_softmax(self.fc3(x), dim =1)

    return x

model = SimpleNN()
print(model)
```

```
SimpleNN(
  (fc1): Linear(in_features=784, out_features=128, bias=True)
  (fc2): Linear(in_features=128, out_features=10, bias=True)
  (fc3): Linear(in_features=10, out_features=1, bias=True)
)
```

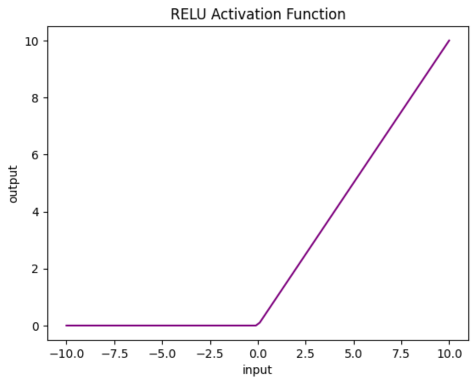
```
#logisitic sigmoid

import torch
import matplotlib.pyplot as plt

x = torch.linspace(-10,10,100)

y = torch.sigmoid(x)

plt.plot(x.numpy(), y.numpy(), color = 'purple')
plt.xlabel('input')
plt.ylabel('output')

plt.title("Logisitic Activation Function")

plt.show()
```



```
#TAnh activation function

y = torch.tanh(x)

plt.plot(x.numpy(), y.numpy(), color = 'purple')
plt.xlabel('input')
plt.ylabel('output')

plt.title("Tanh Activation Function")

plt.show()
```



```
#RELU fucntion

y = torch.relu(x)

plt.plot(x.numpy(), y.numpy(), color = 'purple')
plt.xlabel('input')
plt.ylabel('output')

plt.title("RELU Activation Function")

plt.show()
```



Start coding or generate with AI.