# TCP Server

import socket

def start\_server():

# Create a socket object

server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# Get local machine name

host = socket.gethostname()

port = 9999

# Bind to the port

server\_socket.bind((host, port))

# Listen for incoming connections

server\_socket.listen(1)

print('Server listening on {}:{}'.format(host, port))

while True:

# Wait for a connection

client\_socket, addr = server\_socket.accept()

print('Got a connection from {}'.format(addr))

# Receive data from the client

data = client\_socket.recv(1024).decode()

if not data:

break

print('Received data: {}'.format(data))

# Send back the data to the client

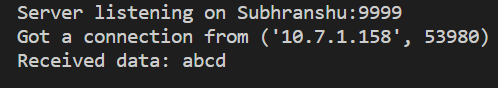
client\_socket.sendall(data.encode())

# Close the connection

client\_socket.close()

if \_\_name\_\_ == '\_\_main\_\_':

start\_server()



import socket

def start\_client():

# Create a socket object

client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# Get local machine name

host = socket.gethostname()

port = 9999

# Connect to the server

client\_socket.connect((host, port))

message = input('Enter message to send: ')

client\_socket.sendall(message.encode())

# Receive data from the server

data = client\_socket.recv(1024).decode()

print('Received data: {}'.format(data))

if \_\_name\_\_ == '\_\_main\_\_':

start\_client()



2.

# TCP Server

import socket

def start\_server():

# Create a socket object

server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# Get local machine name

host = socket.gethostname()

port = 9999

# Bind to the port

server\_socket.bind((host, port))

# Listen for incoming connections

server\_socket.listen(1)

print('Server listening on {}:{}'.format(host, port))

while True:

# Wait for a connection

client\_socket, addr = server\_socket.accept()

print('Got a connection from {}'.format(addr))

# Receive data from the client

data = client\_socket.recv(1024).decode()

if not data:

break

print('Received data: {}'.format(data))

# Echo back the data to the client

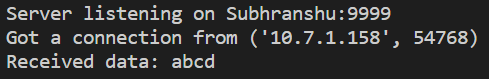
client\_socket.sendall(data.encode())

# Close the connection

client\_socket.close()

if \_\_name\_\_ == '\_\_main\_\_':

start\_server()



import socket

def start\_client():

# Create a socket object

client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# Get local machine name

host = socket.gethostname()

port = 9999

# Connect to the server

client\_socket.connect((host, port))

message = input('Enter message to send: ')

client\_socket.sendall(message.encode())

# Receive data from the server

data = client\_socket.recv(1024).decode()

print('Received data: {}'.format(data))

if \_\_name\_\_ == '\_\_main\_\_':

start\_client()



3.

# TCP Server

import socket

def start\_server():

# Create a socket object

server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# Get local machine name

host = socket.gethostname()

port = 9999

# Bind to the port

server\_socket.bind((host, port))

# Listen for incoming connections

server\_socket.listen(1)

print('Server listening on {}:{}'.format(host, port))

while True:

# Wait for a connection

client\_socket, addr = server\_socket.accept()

print('Got a connection from {}'.format(addr))

while True:

# Receive data from the client

data = client\_socket.recv(1024).decode()

if not data:

break

print('Received data: {}'.format(data))

# Check if the received data is the stop command

if data.lower() == 'stop':

client\_socket.sendall("Closing Connection".encode())

print("Closing connection")

client\_socket.close()

break

# Echo back the data to the client

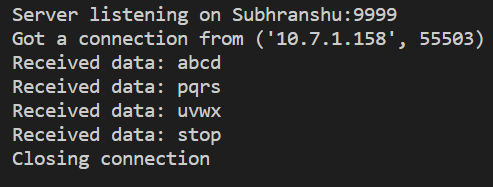
client\_socket.sendall(data.encode())

# Close the connection

client\_socket.close()

if \_\_name\_\_ == '\_\_main\_\_':

start\_server()



# TCP Client

import socket

def start\_client():

# Create a socket object

client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# Get local machine name

host = socket.gethostname()

port = 9999

# Connect to the server

client\_socket.connect((host, port))

while True:

# Get input from user and send it to the server

message = input('Enter message to send: ')

client\_socket.sendall(message.encode())

# Receive data from the server

data = client\_socket.recv(1024).decode()

# Check if the received data is the stop command

if data.lower() == 'stop':

print(client\_socket.recv(1204).decode())

break

print('Received data: {}'.format(data))

# Close the connection

client\_socket.close()

if \_\_name\_\_ == '\_\_main\_\_':

start\_client()

