

REGISTER NUMBER:

Name :

SAUVIK DAS

RA1811028010101

**REGISTER NUMBER:**

**Name :**

<b>Ex : No :</b>	
<b>Date:</b>	

Q 1. Write a python code to create a Simple TCP Client and Server, for the client to get date time from server machine.

Sol.

Server:

```
import socket
import datetime
Date = datetime.datetime.now()
#print(Date)
Sock = socket.socket()
host = socket.gethostname()
port = 9999
Sock.bind((host,port))
print("Connecting")
Sock.listen(10)
while True:
    conn,addr = Sock.accept()
    print('Connected to ',addr)
    conn.send(str(Date).encode())
Sock.close()
```

Client:

```
import socket
Sock = socket.socket()
host = socket.gethostname()
```

**REGISTER NUMBER:**

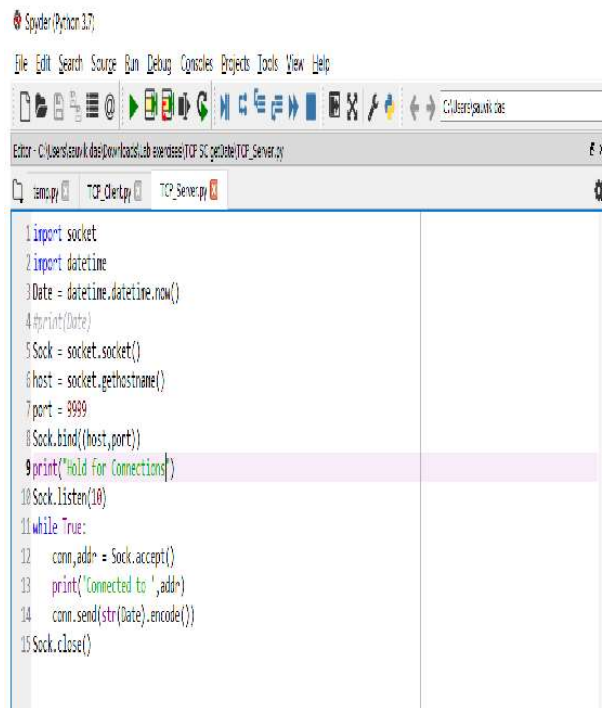
**Name :**

port = 9999

Sock.connect((host,port))

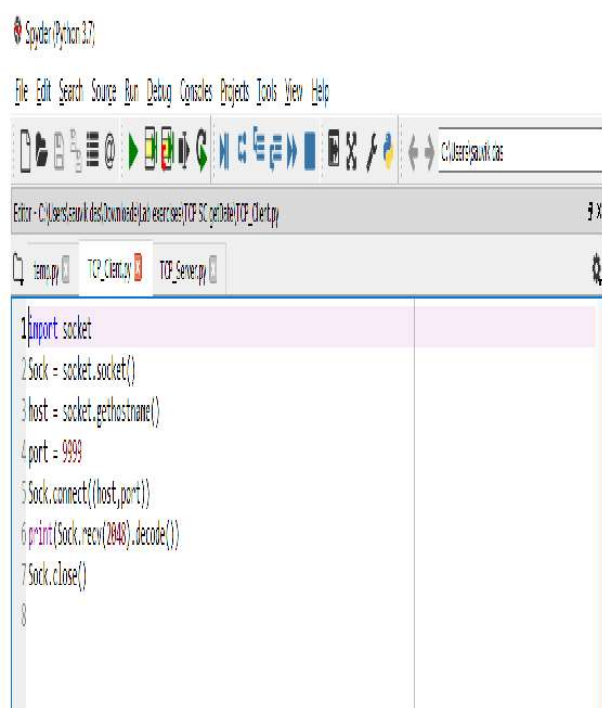
print(Sock.recv(2048).decode())

Sock.close()



The screenshot shows the Spyder Python IDE interface. The file explorer on the left displays a project named 'TCP\_Server.py'. The editor window shows the following Python code:

```
1 import socket
2 import datetime
3 Date = datetime.datetime.now()
4 #print(Date)
5 Sock = socket.socket()
6 host = socket.gethostname()
7 port = 9999
8 Sock.bind((host,port))
9 print("hold for connections")
10 Sock.listen(10)
11 while True:
12     conn,addr = Sock.accept()
13     print('Connected to ',addr)
14     conn.send(str(Date).encode())
15 Sock.close()
```



The screenshot shows the Spyder Python IDE interface. The file explorer on the left displays a project named 'TCP\_Client.py'. The editor window shows the following Python code:

```
1 import socket
2 Sock = socket.socket()
3 host = socket.gethostname()
4 port = 9999
5 Sock.connect((host,port))
6 print(Sock.recv(2048).decode())
7 Sock.close()
8
```

**REGISTER NUMBER:**

**Name :**

Q 2. Write a python code to create Echo client server application using UDP Sockets. The message sent by the client is echoed back to it by the server.

Sol.

Server:

```
import socket

sock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)

host = socket.gethostname()

port = 9999

sock.bind((host,port))

print("Connecting")

while True:

    print("Hold for client")

    data,addr = sock.recvfrom(1024)

    print("Message received !",data.decode()," from",addr)
```

Client:

```
import socket

sock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)

host = socket.gethostname()

port = 9999

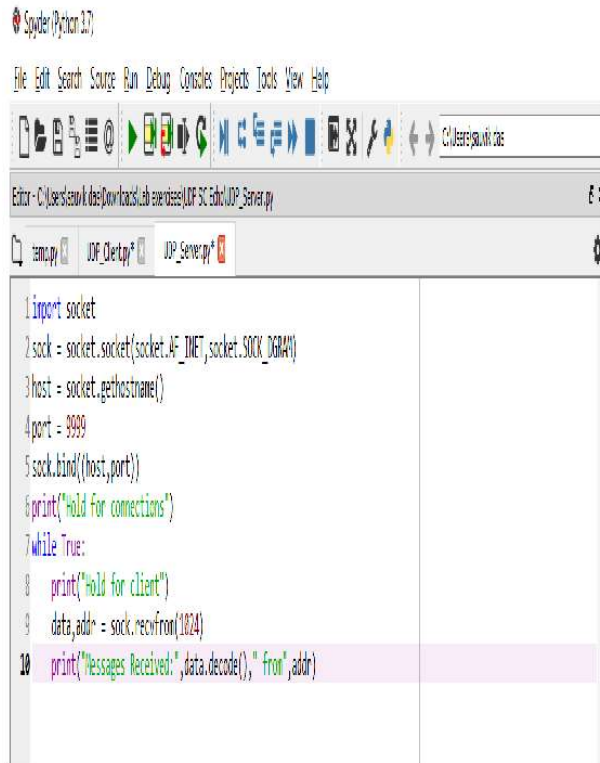
mssg = "This is the Client. The server is Working."

sock.sendto(mssg.encode(),(host,port))

print("Message sent")
```

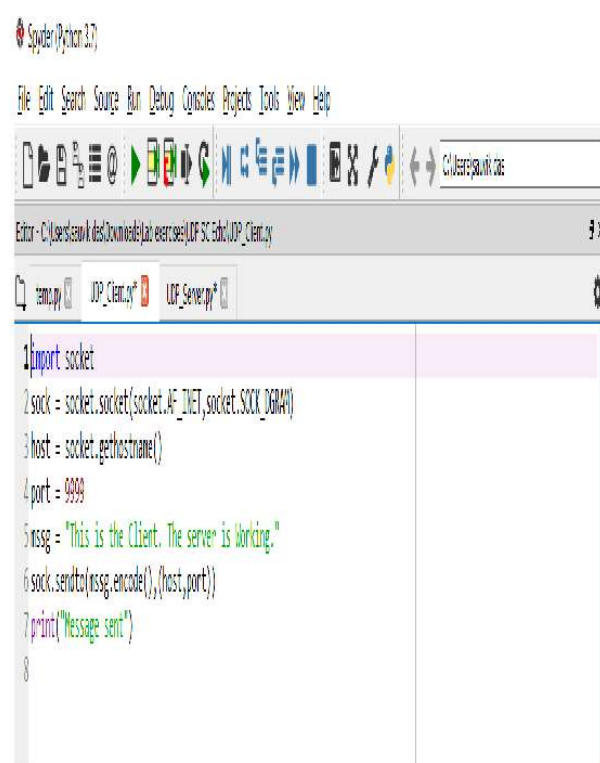
REGISTER NUMBER:

Name :



The screenshot shows a Python IDE with a menu bar (File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, Help) and a toolbar. The editor window is titled 'Editor - C:\Users\saiv\Downloads\lab exercises\UDP SC Echo\UDP\_Server.py'. The file explorer on the left shows a project named 'UDP\_Server' containing 'UDP\_Client.py' and 'UDP\_Server.py'. The code in the editor is as follows:

```
1 import socket
2 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
3 host = socket.gethostname()
4 port = 9999
5 sock.bind((host, port))
6 print("Hold for connections")
7 while True:
8     print("Hold for client")
9     data, addr = sock.recvfrom(1024)
10    print("Messages Received:", data.decode(), " from", addr)
```



The screenshot shows a Python IDE with a menu bar (File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, Help) and a toolbar. The editor window is titled 'Editor - C:\Users\saiv\Downloads\lab exercises\UDP SC Echo\UDP\_Client.py'. The file explorer on the left shows a project named 'UDP\_Server' containing 'UDP\_Client.py' and 'UDP\_Server.py'. The code in the editor is as follows:

```
1 import socket
2 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
3 host = socket.gethostname()
4 port = 9999
5 msg = "This is the Client. The server is working."
6 sock.sendto(msg.encode(), (host, port))
7 print("Message sent")
8
```

**REGISTER NUMBER:**

**Name :**

Q 3. Write a python code to simulate simple client server chat application using TCP/IP Sockets.

Sol.

Server:

```
import socket

sock = socket.socket()

host = socket.gethostname()

port = 9999

sock.bind((host,port))

sock.listen(5)

print("Hold for client")

con,addr = sock.accept()

print("Connected to ",addr)

while True:

    data = con.recv(1024)

    print("Client : "+ data.decode())

    if data == 'quit' or not data:

        print("Server Quitting")

        break

    data = input("Server: ")

    con.sendall(data.encode())

con.close()
```

Client:

```
import socket

import datetime

name = input("Please input your name: ")
```

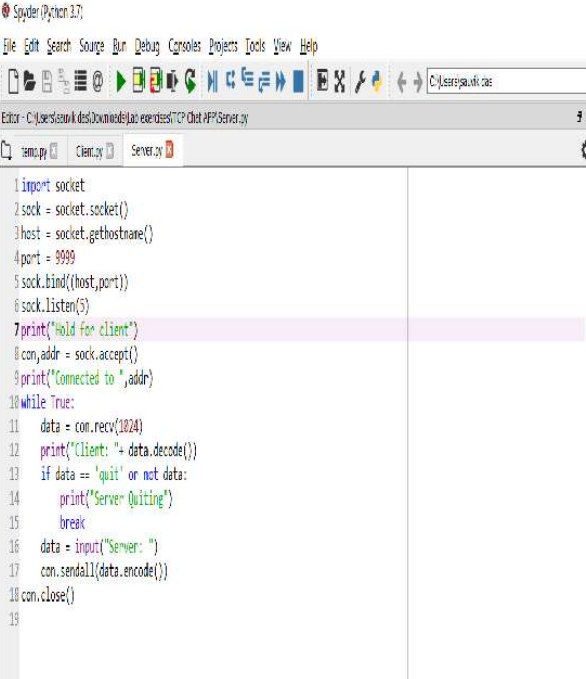
**REGISTER NUMBER:**

**Name :**

```
sock = socket.socket()
host = socket.gethostname()
port = 9999
sock.connect((host,port))
mssg = "You are now connected to " + name + " at " + str(datetime.datetime.now())
print("Server connected")
sock.sendall(mssg.encode())
while True:
    mssg = sock.recv(2048)
    if mssg.decode() == 'quit' or not mssg:
        print(name+" is now disconnecting at "+ str(datetime.datetime.now()))
        break
    else:
        print("Server: "+mssg.decode())
        mssg = input("Client: ")
        sock.sendall(mssg.encode())
sock.close()
```

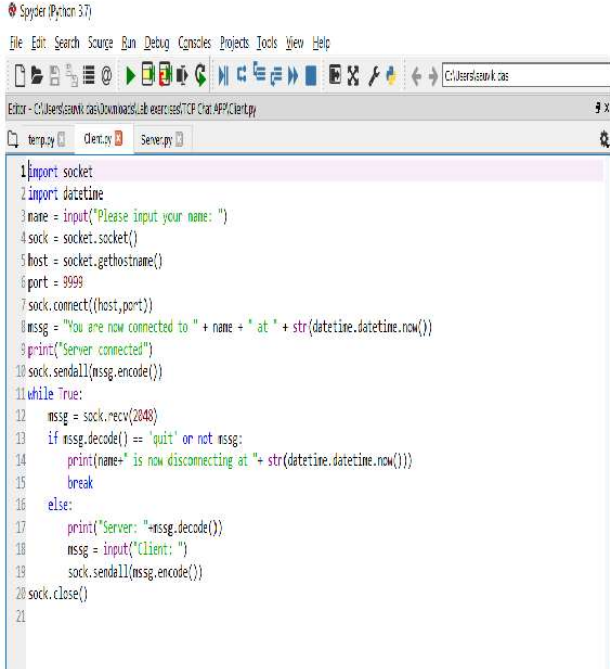
REGISTER NUMBER:

Name :



The screenshot shows the Spyder Python IDE interface. The editor window displays a Python script for a server. The script imports the socket module, creates a socket object, gets the host name, sets the port to 9999, binds the socket to the host and port, and starts listening for connections. It then enters a loop where it accepts connections, prints the client address, and prints 'Hold for client'. The script is saved as C:\Users\ank\d\Downloads\lab exercise\TCP Chat APP\Server.py.

```
1 import socket
2 sock = socket.socket()
3 host = socket.gethostname()
4 port = 9999
5 sock.bind((host,port))
6 sock.listen(5)
7 print("Hold for client")
8 con,addr = sock.accept()
9 print("Connected to ",addr)
10 while True:
11     data = con.recv(1024)
12     print("Client: "+ data.decode())
13     if data == 'quit' or not data:
14         print("Server Quitting")
15         break
16     data = input("Server: ")
17     con.sendall(data.encode())
18 con.close()
19
```



The screenshot shows the Spyder Python IDE interface. The editor window displays a Python script for a client. The script imports the socket and datetime modules, gets the user's name, creates a socket object, gets the host name, sets the port to 9999, and connects to the host and port. It then sends a message to the server, prints the connection status, and enters a loop where it receives messages from the server. If the message is 'quit', it prints the disconnection time and breaks the loop. Otherwise, it prints the received message and sends a response to the server. The script is saved as C:\Users\ank\d\Downloads\lab exercise\TCP Chat APP\Client.py.

```
1 import socket
2 import datetime
3 name = input("Please input your name: ")
4 sock = socket.socket()
5 host = socket.gethostname()
6 port = 9999
7 sock.connect((host,port))
8 msg = "You are now connected to " + name + " at " + str(datetime.datetime.now())
9 print("Server connected")
10 sock.sendall(msg.encode())
11 while True:
12     msg = sock.recv(2048)
13     if msg.decode() == 'quit' or not msg:
14         print(name+" is now disconnecting at "+ str(datetime.datetime.now()))
15         break
16     else:
17         print("Server: "+msg.decode())
18         msg = input("Client: ")
19         sock.sendall(msg.encode())
20 sock.close()
21
```



**REGISTER NUMBER:**

**Name :**

Q 4. Write a python code to get the IP address of the client machine using UDP sockets.

Sol.

Server:

```
import socket

host = socket.gethostname()

port = 9999

sock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)

sock.bind((host,port))

print("Hold for client")

while True:

    data, addr = sock.recvfrom(2048)

    print('Client Ip address :'+data.decode())

    print("Server Exiting")

    sock.sendto("Thanks" .encode(),(host,port))

    break

sock.close()
```

Client:

```
import socket

import socket

host = socket.gethostname()

port = 9999

sock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)

hostname = socket.gethostname()

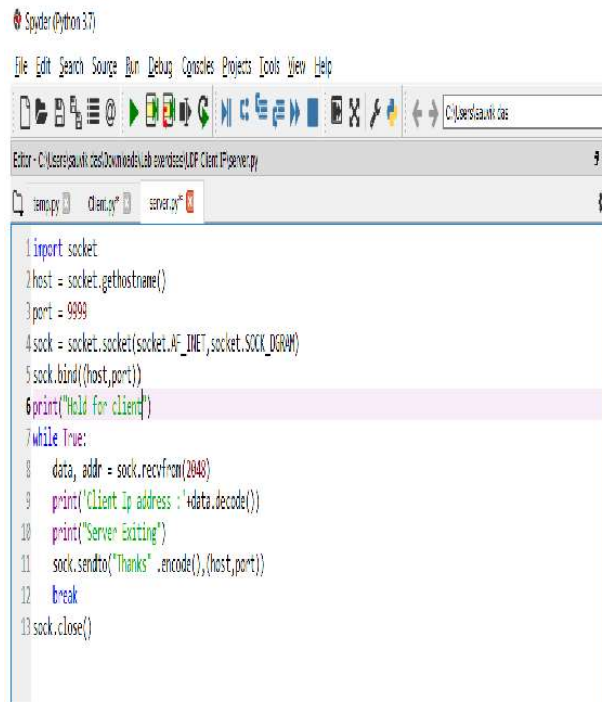
IPAddr = str(socket.gethostbyname(hostname))

sock.sendto(IPAddr.encode(),(host,port))
```

REGISTER NUMBER:

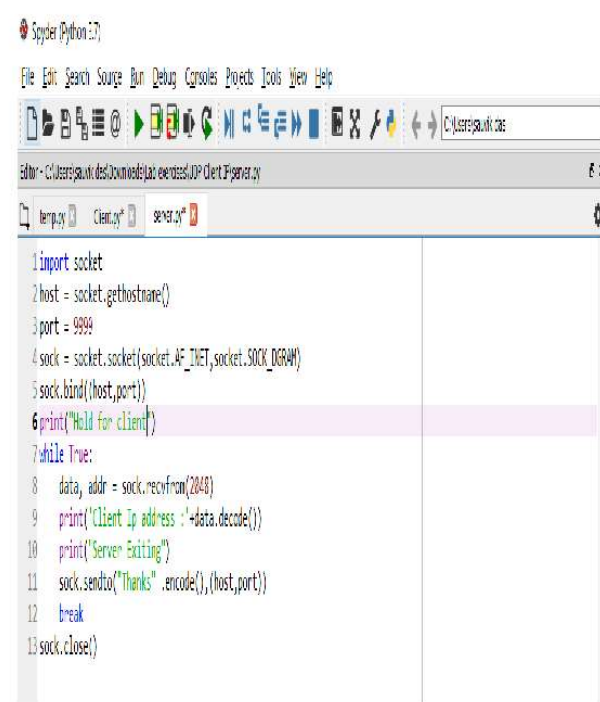
Name :

print("IP Sent to Server")



The screenshot shows the Spyder Python IDE interface. The file explorer on the left displays a project named 'server.py'. The main editor window shows the following Python code:

```
1 import socket
2 host = socket.gethostname()
3 port = 9999
4 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
5 sock.bind((host, port))
6 print("Hold for client")
7 while True:
8     data, addr = sock.recvfrom(2048)
9     print('Client Ip address :'+data.decode())
10    print('Server Exiting')
11    sock.sendto("Thanks".encode(), (host, port))
12    break
13 sock.close()
```



The screenshot shows the Spyder Python IDE interface. The file explorer on the left displays a project named 'client.py'. The main editor window shows the following Python code:

```
1 import socket
2 host = socket.gethostname()
3 port = 9999
4 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
5 sock.bind((host, port))
6 print("Hold for client")
7 while True:
8     data, addr = sock.recvfrom(2048)
9     print('Client Ip address :'+data.decode())
10    print('Server Exiting')
11    sock.sendto("Thanks".encode(), (host, port))
12    break
13 sock.close()
```

**REGISTER NUMBER:**

**Name :**

Q 5. Create a Simple multi point client server chat using TCP Sockets.

Sol.

Server:

```
import socket
from threading import Thread
def threads():
    while True:
        data = conn.recv(1024)
        print('Client Request :' + data.decode())
        if data == 'quit' or not data:
            print("Server Exiting")
            break
        data = input('Server Response:')
        conn.sendall(data.encode())
    host = socket.gethostname()
    port = 9999
    s = socket.socket()
    s.bind((host,port))
    s.listen(10)
    print("Hold For Clients")
    while True:
        conn,addr = s.accept()
        print("Connected to ", addr)
        child = Thread(target=threads)
        child.start()
    conn.close()
```

**REGISTER NUMBER:**

**Name :**

Client:

```
import socket
```

```
import datetime
```

```
name = input("Enter Client Name : ")
```

```
sock = socket.socket()
```

```
host = socket.gethostname()
```

```
port = 9999
```

```
sock.connect((host,port))
```

```
mssg = "You are now connected to " + name + " at " + str(datetime.datetime.now())
```

```
print("Server connected")
```

```
sock.sendall(mssg.encode())
```

```
while True:
```

```
mssg = sock.recv(2048)
```

```
if mssg.decode() == 'quit' or not mssg:
```

```
print(name+"is now Disconnecting at "+ str(datetime.datetime.now()))
```

```
break
```

```
else:
```

```
print("Server: "+mssg.decode())
```

```
mssg = input("Client: ")
```

```
sock.sendall(mssg.encode())
```

```
sock.close()
```

REGISTER NUMBER:

Name :

```

Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\saun\Documents\Downloads\Lib\examples\TCP_Multichat\APPClient.py
server.py
1 import socket
2 from threading import Thread
3 def threads():
4     while True:
5         data = conn.recv(1024)
6         print('Client Request : ' + data.decode())
7         if data == 'quit' or not data:
8             print("Server Exiting")
9             break
10        data = input('Server Response:')
11        conn.sendall(data.encode())
12
13
14 host = socket.gethostname()
15 port = 9999
16 s = socket.socket()
17 s.bind((host,port))
18 s.listen(10)
19 print("Hold For Clients")
20 while True:
21     conn,addr = s.accept()
22     print("Connected by ", addr)
23     child = Thread(target=threads)
24     child.start()
25 conn.close()

```

```

Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\saun\Documents\Downloads\Lib\examples\TCP_Multichat\APPClient.py
temp.py
1 import socket
2 import datetime
3 name = input("Enter Client Name : ")
4 sock = socket.socket()
5 host = socket.gethostname()
6 port = 9999
7 sock.connect((host,port))
8 nmsg = "You are now connected to " + name + " at " + str(datetime.datetime.now())
9 print("Server connected")
10 sock.sendall(nmsg.encode())
11 while True:
12     msg = sock.recv(2048)
13     if msg.decode() == 'quit' or not msg:
14         print(name+"Is now Disconnecting at "+ str(datetime.datetime.now()))
15         break
16     else:
17         print("Server: " +msg.decode())
18         nmsg = input("Client: ")
19         sock.sendall(nmsg.encode())
20 sock.close()
21

```

**REGISTER NUMBER:**

**Name :**

Q 6. Write a Python code to Implement the File Transfer Protocol using TCP Sockets. (FTP Example in common classroom).

Sol.

Server:

```
import socket

host = socket.gethostname()

port = 9999

sock = socket.socket()

sock.bind((host,port))

sock.listen(10)

print("Hold for connections")

while True:

    conn, addr = sock.accept()

    print("Connected to ",addr)

    data = conn.recv(2048)

    print("Server recieved : " ,data.decode())

    file = open("input.txt",'rb')

    l = file.read(1024)

    while(l):

        conn.send(l)

        print('sent',l)

        l = file.read(1024)

    file.close()

    print("File Sent!")

    conn.close()
```

**REGISTER NUMBER:**

**Name :**

Client:

```
import socket

host = socket.gethostname()

port =9999

sock = socket.socket()

sock.connect((host,port))

sock.send("Cleint Connected.".encode())

with open('output.txt','wb') as file:

print('File Created.')

while True:

print('Recieving Data...')

data = sock.recv(1024)

print('Data Recieved : ', data.decode())

if not data:

break

file.write(data)

file.close()

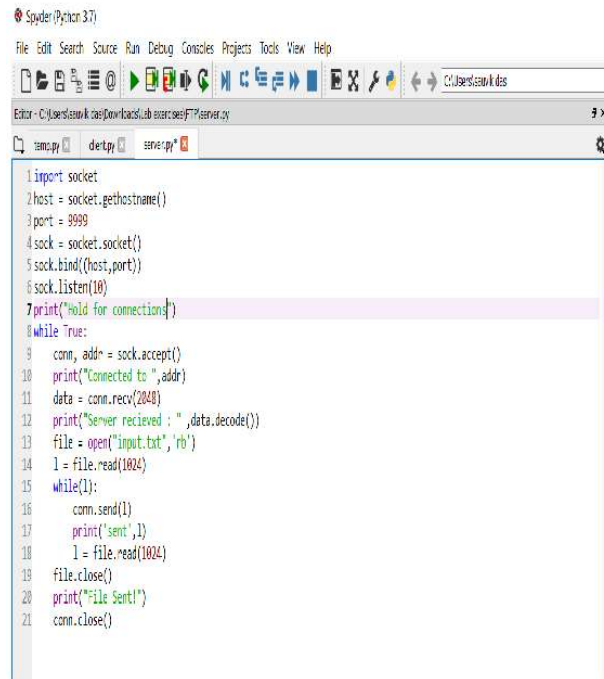
print("Write Successful")

sock.close()

print("Client Disconnected")
```

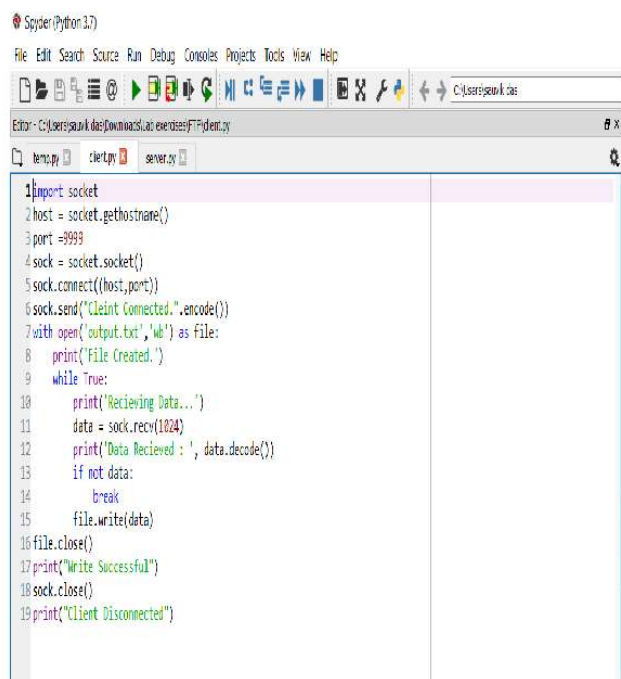
REGISTER NUMBER:

Name :



The screenshot shows the Spyder Python IDE interface. The editor window displays the code for 'server.py'. The code is as follows:

```
1 import socket
2 host = socket.gethostname()
3 port = 9999
4 sock = socket.socket()
5 sock.bind((host,port))
6 sock.listen(10)
7 print("hold for connection")
8 while True:
9     conn, addr = sock.accept()
10    print("Connected to ", addr)
11    data = conn.recv(2048)
12    print("Server recieved : ", data.decode())
13    file = open("input.txt", 'rb')
14    l = file.read(1024)
15    while(l):
16        conn.send(l)
17        print('sent', l)
18        l = file.read(1024)
19    file.close()
20    print("file Sent!")
21    conn.close()
```



The screenshot shows the Spyder Python IDE interface. The editor window displays the code for 'client.py'. The code is as follows:

```
1 import socket
2 host = socket.gethostname()
3 port = 9999
4 sock = socket.socket()
5 sock.connect((host,port))
6 sock.send("Client Connected.".encode())
7 with open('output.txt', 'w') as file:
8     print('File Created.')
9     while True:
10        print('Receiving Data...')
11        data = sock.recv(1024)
12        print('Data Received : ', data.decode())
13        if not data:
14            break
15        file.write(data)
16    file.close()
17 print("Write Successful")
18 sock.close()
19 print("Client Disconnected")
```



**REGISTER NUMBER:**

**Name :**

Q 7. Create a server application which send the Ip address to client for the given domain name using UDP and use dictionary to store the domain and IP address combination. (DNS Server Example in common classroom).

Sol.

Server:

```
import socket

dns = {
    'google': '172.13.31.89',
    'gmail': '154.32.176.11',
    'w3school': '143.76.142.21',
    'yahoo': '162.54.26.11',
    'bing': '148.47.25.33',
    'youtube': '163.38.31.11'
}

host = socket.gethostname()

sock_r = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock_s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock_r.bind((host,9999))

print("Waiting for client...")

while True:
    data, addr = sock_r.recvfrom(1024)
    print('Domain :'+data.decode(), addr)
    ip = ""
    if data.decode() in dns.keys():
        ip = dns.get(data.decode())
    elif( data.decode() == "bye" ) or not data:
        break
```

**REGISTER NUMBER:**

**Name :**

else:

```
ip = "Not Found"
```

```
sock_s.sendto(ip.encode(),(host,9998))
```

```
sock_r.close()
```

```
sock_s.close()
```

Client:

```
import socket
```

```
host = socket.gethostname()
```

```
sock_r = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
```

```
sock_s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
```

```
sock_r.bind((host, 9998))
```

```
print("Connecting...")
```

```
while True:
```

```
data = input("Domain : ")
```

```
if data == 'bye' or not data:
```

```
break
```

```
else:
```

```
sock_s.sendto(data.encode(), (host, 9999))
```

```
data = sock_r.recv(1024)
```

```
print("IP: "+data.decode())
```

```
sock_s.close()
```

```
sock_r.close()
```

REGISTER NUMBER:

Name :

```
Spider(Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\punik\Desktop\IP Domain\server.py
temp.py Client.py server.py
1 import socket
2 dns = {
3     'google': '172.13.31.89',
4     'gmail': '156.32.176.11',
5     'w3school': '143.76.142.21',
6     'yahoo': '162.54.26.11',
7     'bing': '148.47.25.39',
8     'youtube': '163.30.31.11'
9 }
10 host = socket.gethostname()
11 sock_r = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
12 sock_s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
13 sock_r.bind((host, 9999))
14 print("Waiting for client...")
15 while True:
16     data, addr = sock_r.recvfrom(1024)
17     print('Domain :'+data.decode(), addr)
18     ip = ""
19     if data.decode() in dns.keys():
20         ip = dns.get(data.decode())
21     elif data.decode() == "bye" or not data:
22         break
23     else:
24         ip = "Not Found"
25     sock_s.sendto(ip.encode(), (host, 9999))
26 sock_r.close()
27 sock_s.close()
28
```

```
Spider(Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\punik\Desktop\IP Domain\client.py
temp.py Client.py server.py
1 import socket
2 host = socket.gethostname()
3 sock_r = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
4 sock_s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
5 sock_r.bind((host, 9999))
6 print("Connecting...")
7 while True:
8     data = input("Domain : ")
9     if data == "bye" or not data:
10         break
11     else:
12         sock_s.sendto(data.encode(), (host, 9999))
13         data = sock_r.recv(1024)
14         print("IP: "+data.decode())
15 sock_s.close()
16 sock_r.close()
17
```