Networking

What is networking?

Networking is logical or physical link between one or more devices. The advantage of networking is sharing resources (hardware/software).

In networking there are two applications

1. Client
2. Server

Client program send message to server program.
Server program receives this message and process it generate response or output.

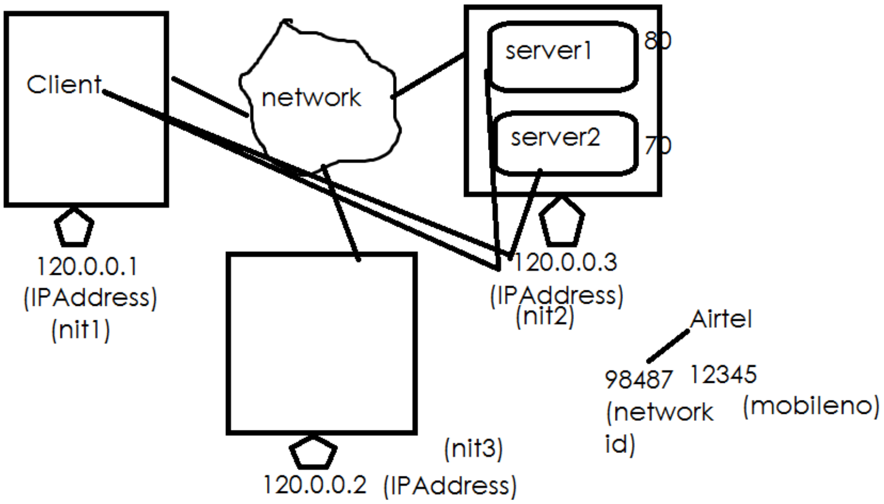
Python provides a predefined module or library called “socket” module. Using “socket” module we can develop network applications.

Q: What is ip-address?
In networking each system is identified with one unique number called ip-address. The ip-address is given by network admins.

Q: What is hostname?
Hostname is a wrapper of ip-address.

Q: What is protocol?
Protocol defines set of rules and regulations to communicate one application with another application.
Example: HTTP, FTP, SMTP, POP,…

Q: What is portno?
Portno is an integer number to identify each program in network.
Portno is 16 bit integer number (1- 65535)



What is socket?
socket is an endpoint communication between two program (OR) socket is an implementation of client or server.
Socket implement specific protocol.

There are two types of sockets provided by socket module

1. TCP implementation
2. UDP implementation

TCP stands for Transmission control protocol, it is a connection oriented protocol.
UDP stands for User Datagram Protocol, it is a connection less protocol.

Socket class

class socket.socket(family=AF_INET, type=SOCK_STREAM)
Create a new socket using the given address family, socket type and protocol number
Default socket type is SOCK_STREAM which is TCP implementation.

socket.accept()
Accept a connection. The socket must be bound to an address and listening for connections. The return value is a pair (conn, address) where *conn* is a *new* socket object usable to send and receive data on the

connection, and *address* is the address bound to the socket on the other end of the connection.

socket.bind(*address*)
Bind the socket to *address*. The socket must not already be bound.

socket.close()
Mark the socket closed

socket.send(*bytes*[, *flags*])
Send data to the socket. The socket must be connected to a remote socket

socket.recv(*bufsize*[, *flags*])
Receive data from the socket. The return value is a bytes object representing the data received.

socket.listen([*backlog*])
Enable a server to accept connections.

Server program

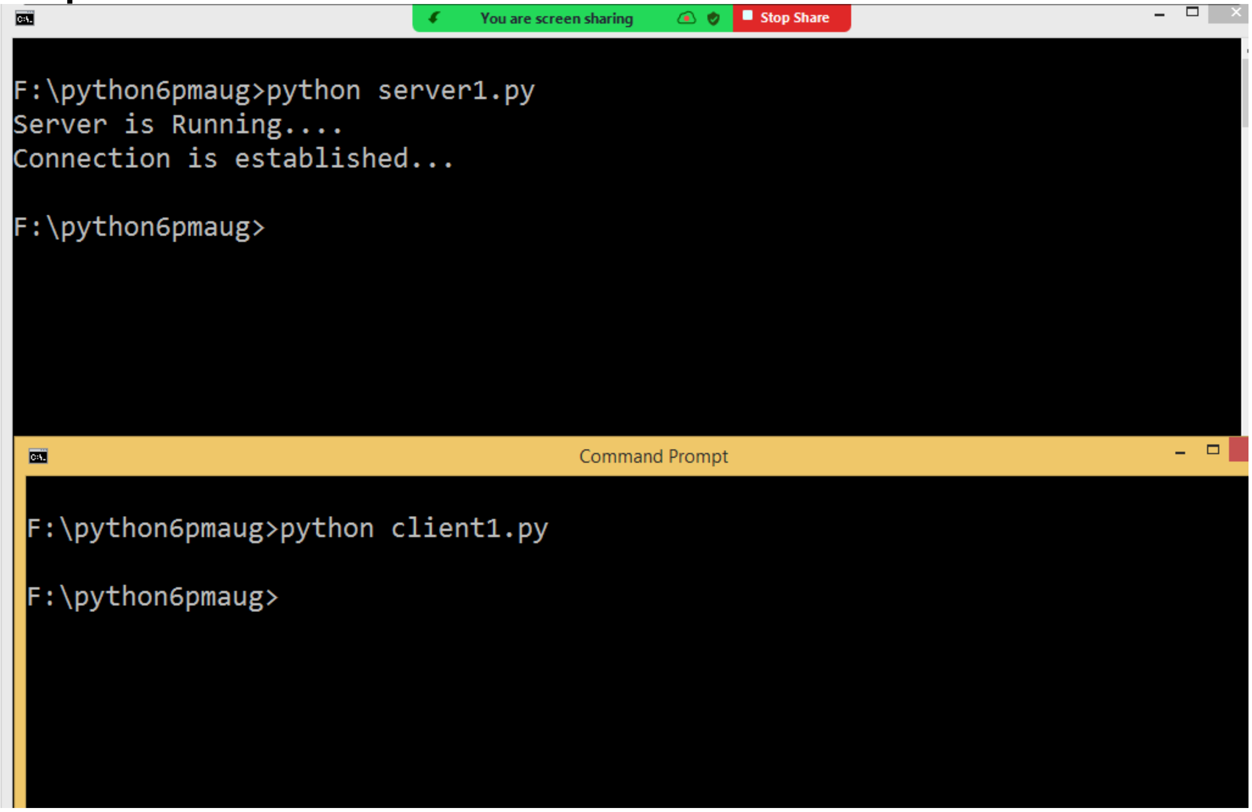
```
# Server Program
import socket
def main():
    s=socket.socket()
    s.bind(("localhost",30))
    s.listen(5)
    print("Server is Running....")
    c=s.accept()
    print("Connection is established...")

main()
```

Client Program

```
# client program
import socket
def main():
    s=socket.socket()
    s.connect(("localhost",30))
```

main()

Output:



```
F:\python6pmaug>python server1.py
Server is Running....
Connection is established...

F:\python6pmaug>
```

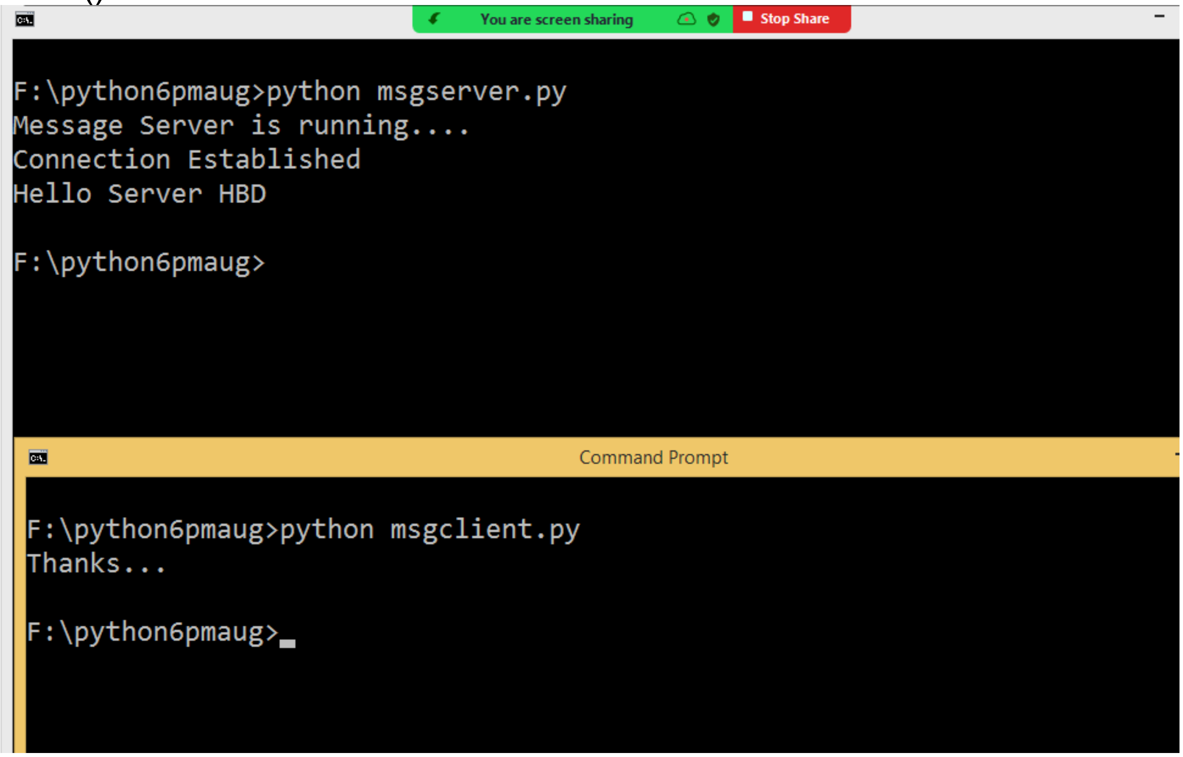
```
F:\python6pmaug>python client1.py

F:\python6pmaug>
```

Example:
Message Server
# Message Server

```
import socket
def main():
    s=socket.socket()
    s.bind(("localhost",40))
    s.listen(10)
    print("Message Server is running....")
    c=s.accept()
    print("Connection Established")
    cs=c[0]
    b=cs.recv(1024)
    msg=b.decode()
    print(msg)
    msg="Thanks..."
    cs.send(msg.encode())

main()
```

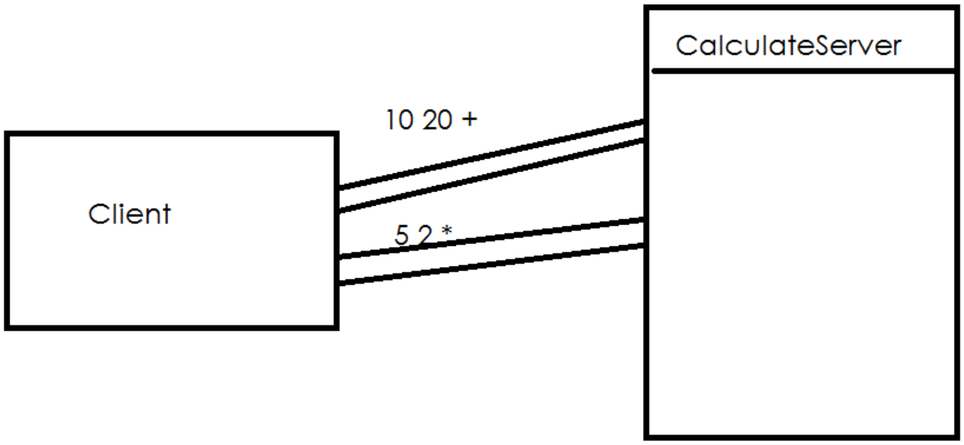
Message Client

```
# message client
import socket
def main():
    s=socket.socket()
    s.connect(("localhost",40))
    msg="Hello Server HBD"
    b=msg.encode()
    s.send(b)
    b=s.recv(1024)
    msg=b.decode()
    print(msg)

main()
```



```
F:\python6pmaug>python msgserver.py
Message Server is running....
Connection Established
Hello Server HBD

F:\python6pmaug>
```

```
F:\python6pmaug>python msgclient.py
Thanks...

F:\python6pmaug>_
```



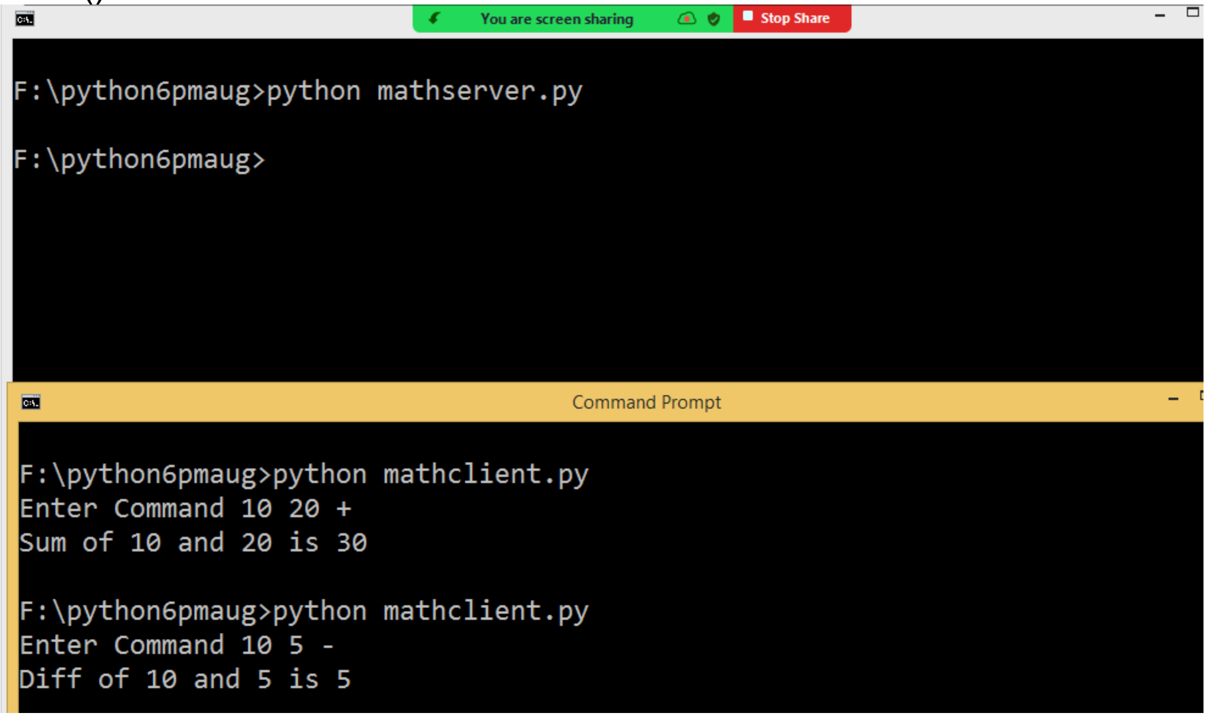
Example:

```
# calculator server
import socket
def main():
    s=socket.socket()
    s.bind(("localhost",40))
    s.listen(5)
    while True:
        c=s.accept()
        cs=c[0]
        b=cs.recv(1024)
        cmd=b.decode()
        o1,o2,opr=cmd.split()
        if opr=='+':
            output=f'Sum of {o1} and {o2} is {int(o1)+int(o2)}'
        elif opr=='-':
            output=f'Diff of {o1} and {o2} is {int(o1)-int(o2)}'
        elif opr=='*':
            output=f'Product of {o1} and {o2} is {int(o1)*int(o2)}'
        elif opr=='/':
            output=f'Div of {o1} and {o2} is {int(o1)/int(o2)}'
        else:
            break
        b=output.encode()
        cs.send(b)
```

```
main()
```

# client

```
import socket
def main():
    s=socket.socket()
    s.connect(("localhost",40))
    cmd=input("Enter Command ")
    b=cmd.encode()
    s.send(b)
    b=s.recv(1024)
    output=b.decode()
    print(output)

main()
```



```
F:\python6pmaug>python mathserver.py

F:\python6pmaug>
```

```
F:\python6pmaug>python mathclient.py
Enter Command 10 20 +
Sum of 10 and 20 is 30

F:\python6pmaug>python mathclient.py
Enter Command 10 5 -
Diff of 10 and 5 is 5
```