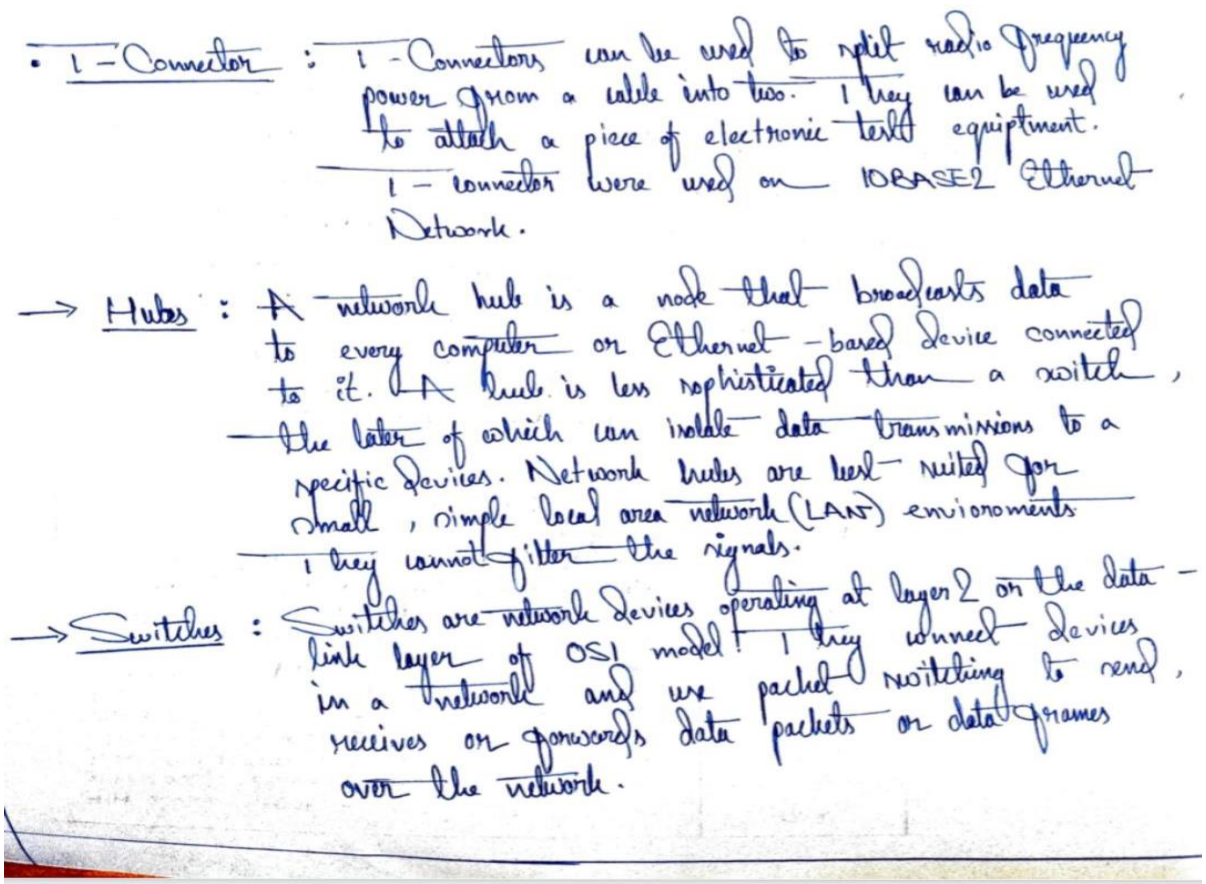
<u>Command prompt – CN (day-1)</u>

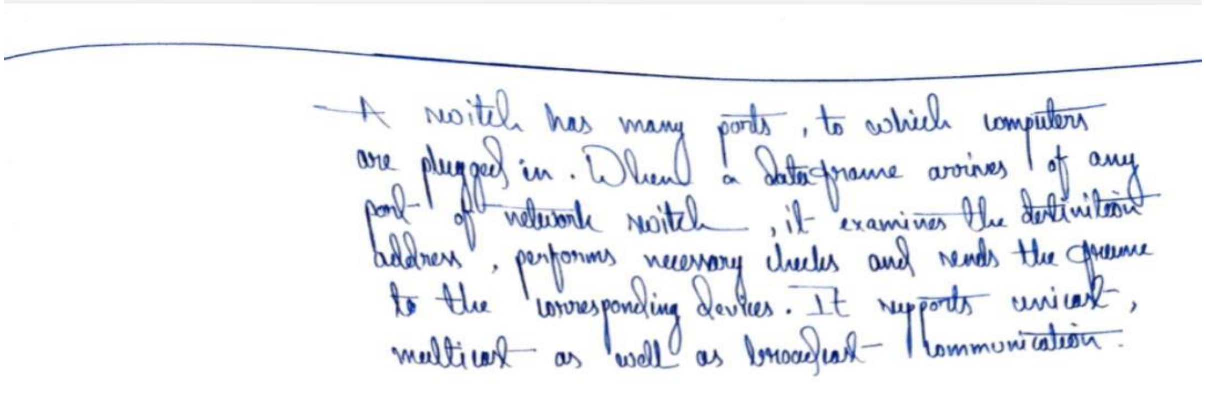
1. **ipconfig –** display info abt host
2. **ipconfig/all –** detailed config abt tcp/ip
3. **nslookup –** daignsng DNS problem
4. **ping <ip> -** sends 1 datagram per sec and prints 1line of outpt for evry response received . Calclts round trip time and displays brief summry
5. **tracert <ip> -** displays all routers that a packet has to go thru
6. **netstat –** displays variety of abt a comp tcp/ip connctn



• T-Connector : T-Connectors can be used to split radio frequency power from a cable into two. They can be used to attach a piece of electronic test equipment.
— T-connector were used on 10BASE2 Ethernet Network.

→ Hubs : A network hub is a node that broadcasts data to every computer or Ethernet-based device connected to it. A hub is less sophisticated than a switch, the later of which can isolate data transmissions to a specific devices. Network hubs are best suited for small, simple local area network (LAN) environments.
— They cannot filter the signals.

→ Switches : Switches are network devices operating at layer 2 or the data-link layer of OSI model. They connect devices in a network and use packet switching to send, receives or forwards data packets or data frames over the network.



— A switch has many ports, to which computers are plugged in. When a data frame arrives at any port of network switch, it examines the destination address, performs necessary checks and sends the frame to the corresponding devices. It supports unicast, multicast as well as broadcast communication.

NETWORKING CABLES

• CAT5 – Category 5 cable is twisted pair cable for carrying signals. This type of cable used in structured cabling for computer networks such as Ethernet. The cable provides performance of upto 100 MHz and is suitable for 10BASE-T, 100BASE-TX (Fast Ethernet) and 1000BASE-T (Gigabit Ethernet). CAT5 is also used to carry other signals such as telephony and video.

• UTP – Unshielded twisted pairs, a UTP cable is used in computer networking that consist of two shielded wires twisted around each other. As the name would imply, these cables do not have insulations (shielding) between each of the paired wires. Consequently, they do not block electromagnetic interference, resulting in a higher risk of packet loss or corruption. Used in Ethernet cables and telephone lines.

Category	Use (Bandwidth)
1	Analog voice grade (0.4 MHz)
2	Older terminal systems up to 4 Mbps (4 MHz)
3	Digital Transmission up to 10 Mbps (16 MHz)
4	" " " " 16 Mbps (20 MHz)
5	" " " " 100 Mbps (100 MHz)
5e	" " " " 1 GBps (100 MHz)
6	" " " " 10 Gbps (250 MHz)
6a	" " " " 10 Gbps (500 MHz)
7	" " " " 10 Gbps (600 MHz)
7a	" " " " 10 Gbps (1000 MHz)
8	" " " " 40 Gbps (2000 MHz)
8.2	" " " " 40 Gbps (2000 MHz)

→ Connectors

• RJ45 : A registered jack (RJ) is a standard physical network interface for connecting telecommunication or data equiptment. The physical connector that registered jacks used are mainly of the modular connector and 50-pin miniature ribbon connector types.
The most common twisted-pair connector is an 8-position, 8-contact (8P8C) modular plug and jack commonly referred to as an RJ45 connector.

Assignment -2 (Week 2)

Cisco Packet Tracer Lab - Basic Switch Configuration | Hostname | Password.

① Configure the host name of the switch as SW1.

```
Switch>ENABLE
Switch#CONFIG T
Enter configuration commands, one per line. End with CNTL/Z.
Switch(config)#hostname SW1
SW1(config)#S.
```

② Set a message of the day (MOTD) banner for the switch -

```
***************
Only Authorized Users Allowed
****************

SW1 (config)#banner ?
    motd  Set Message of the Day banner
SW1 (config)#          banner motd ?
    LINE  c banner-text c, where 'c' is a delimiting character
SW1 (config) #banner motd $ ***********
Enter TEXT message. End with the character '$'.
    *********************
        Only Authorized Users Allowed
    *********************
    $
SW1 (config)#exit              (To check)
SW1 #
%SYS-5-CONFIG_I: Configured from console by console
exit
```

```
SW1 con0 is now available

Press RETURN to get started.

*********
************************
    Only Authorized Users Allowed
**************************
```

③ ⓘ Configure a line console password - India@123

```
SW1>en
SW1# sh run
    Building configuration....
    Current configuration : 1211 bytes
    !
    version 12.2
    no service timestamps log datetime msec
    no service timestamps debug datetime msec
    no service password-encryption
    !
    hostname SW1
    !
    !
    :
    :
    :

SW1# config t
Enter configuration commands, one per line. End with CNTL/Z
SW1(config)# line con0
~~SW1(config)-~~ SW1(config-line)# password India@123
```

ⓘⓘ configure an enable secret password - Venu@123

```
SW1(config)#exit
SW1#
%SYS-5-CONFIG_I : Configured from console by console
SW1# config t
```

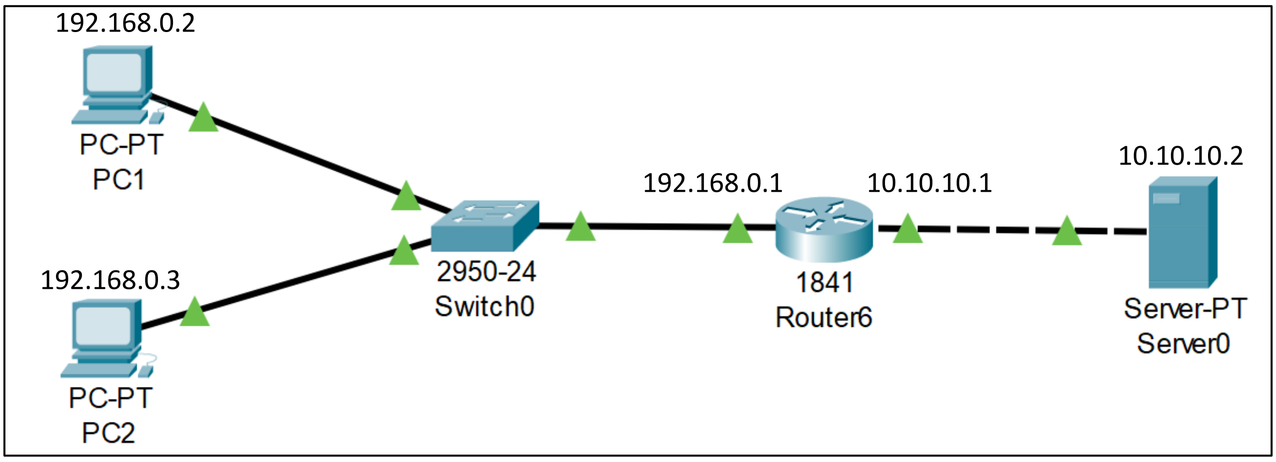
```
Enter configuration commands, one per line. END with CNTL/Z.
SW1(config)#enable secret Venu@123.
SW1(config)#line con 0          (TO Check)
SW1(config-line)#login

***************
    ~~User~~
    Only Authorized User Allowed
*******************

User Access Verification
Password: India@123
~~SW1(config~~
Press Return to get started
SW1>enable
 Password: Venu@123
SW1#config t
```

Assignment - 03

- **FTP Server Configuration:**

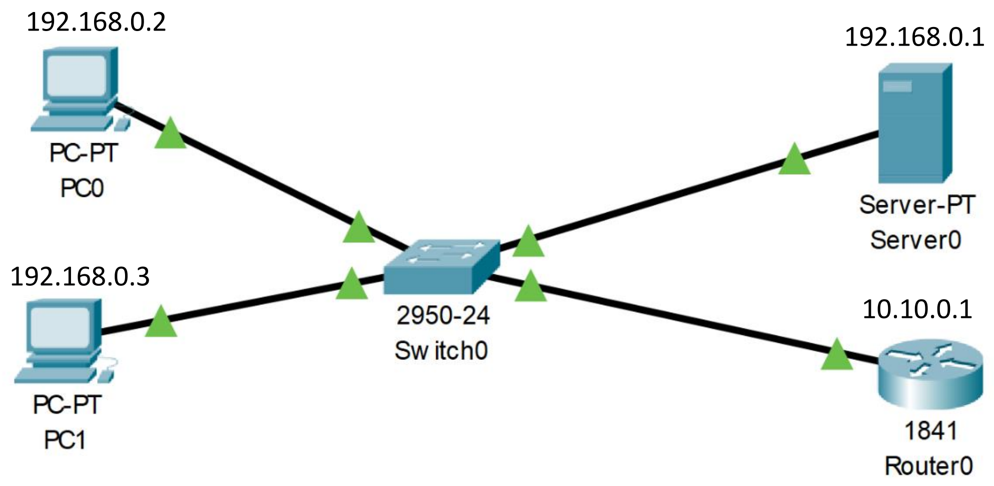
Steps:

1) Open Cisco Packet Tracer and select 2 End Devices (PC device), 1 Switch, 1 Router, 1 Server.
2) Now Connect all the devices using the auto connection.
3) Then configure the IP addresses as per the diagram.



4) Now just wait for some time to let all the connection status turns green.
5) Now we have achieved a connection where a class C IP address is being translated to class A IP Address.
6) Go to one of the PC devices and on Desktop tab select CMD.
7) Now we need to check the connection to the server by `C:\>ping 10.10.10.2`
8) If reply is coming then it means the server is properly configured and connected.
9) Go to the Server → Services →FTP.
10) Put on the FTP service and give username and password and click on ADD.
11) Come back to PC device and open the CMD and type `C:\>ftp 10.10.10.2`
12) It will ask for username and password. Provide the username and password configured earlier.
13) Once the connection is established exit rom the CMD and go to Text Editor and make a new text file.
14) Save the new text file and return to cmd and type `ftp>put filenme.txt`
15) This will send the text file from the PC device (192.168.0.2) to Server (10.10.10.2).
16) Now to verify that the file has been transferred to the server, so type `ftp>dir`
17) You will see your Filename in the list.
18) Now to get a file from server to PC type `ftp>get filename.txt`
19) Now exit from FTP type ctrl+C, then type dir to check that the file is there in the PC or not.
20) So we have successfully send and got a file from a server using FTP protocol.

- **DHCP Server Configuration:**
  1) Open Cisco Packet Tracer and select 2 End Devices (PC device), 1 Switch, 1 Router, 1 Server.
  2) Now Connect all the devices using the auto connection.
  3) Then configure the IP addresses as per the diagram.



  4) Now just wait for some time to let all the connection status turns green.
  5) Now go to the server → Desktop → IP Configuration and set the IP 192.168.0.1.
  6) Go to Services → DHCP and set Default Gateway 192.168.0.1 and DNS Server 10.0.0.1.
  7) Set the Start IP 192.168.0.0 and Max Users 256
  8) Now go to every PC device → Desktop → IP Configuration and set it to DHCP.
  9) Now all the PC will have a DHCP address.

Assignment - 04

1) **Write a program to find the IP address of the system.**

```
import socket
print("Host Name:", socket.gethostname(), "\nIp
Address:", socket.gethostbyname(socket.gethostname()))
```

2) **Write a socket program for implementation of echo.**

Server side code:

```
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind(("127.0.0.1", 9090))
s.listen()
(c, cip) = s.accept()
c.send(c.recv(1024))
s.close()
```

Client side code:

```
import socket
c = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
c.connect(("127.0.0.1", 9090))
data = input()
c.send(data.encode())
dataFromServer = c.recv(1024)
print(dataFromServer.decode())
c.close()
```

3) **Write a client-server application for chat using TCP.**

Server side code:

```
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind(("127.0.0.1", 9090))
s.listen()
while True:
    (c, cip) = s.accept()
    data = c.recv(1024).decode()
    print("Client:",data)
    data = input("Enter Text: ")
    c.send(data.encode())
```

Client side code:

```python
import socket
c = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
c.connect(("127.0.0.1", 9090))
while True:
    data = input("Enter Text: ")
    c.send(data.encode())
    data = c.recv(1024).decode()
    print("Server:",data)
```

4) **Write a program using client server socket programming: Client needs to authenticate itself by entering a server defined string as a password (like OTP) and then to say Hi to server. Server replies with a Hello. Press any key to exit.**

Server side code:

```python
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind(("127.0.0.1", 9090))
s.listen()
(c, cip) = s.accept()
c.send("Enter OTP:".encode())
otp = c.recv(1024).decode()
if otp == '8894':
    c.send("You are Authenticated".encode())
    data = c.recv(1024).decode()
    print("Client:",data)
    data = input("Enter Text: ")
    c.send(data.encode())
else:
    c.send("You are Authenticated".encode())
s.close()
```

Client side code:

```python
import socket
c = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
c.connect(("127.0.0.1", 9090))
data = c.recv(1024).decode()
print(data, end=" ")
otp = input()
c.send(otp.encode())
data = c.recv(1024).decode()
print(data)
```

```python
if data == "You are Authenticated":
    data = input("Enter Text: ")
    c.send(data.encode())
    data = c.recv(1024).decode()
    print("Server:",data)
else:
    c.close()
```

Assignment - 05

1) **Write a program to Perform File Transfer in Client & Server Using TCP/IP.**

Server side code:

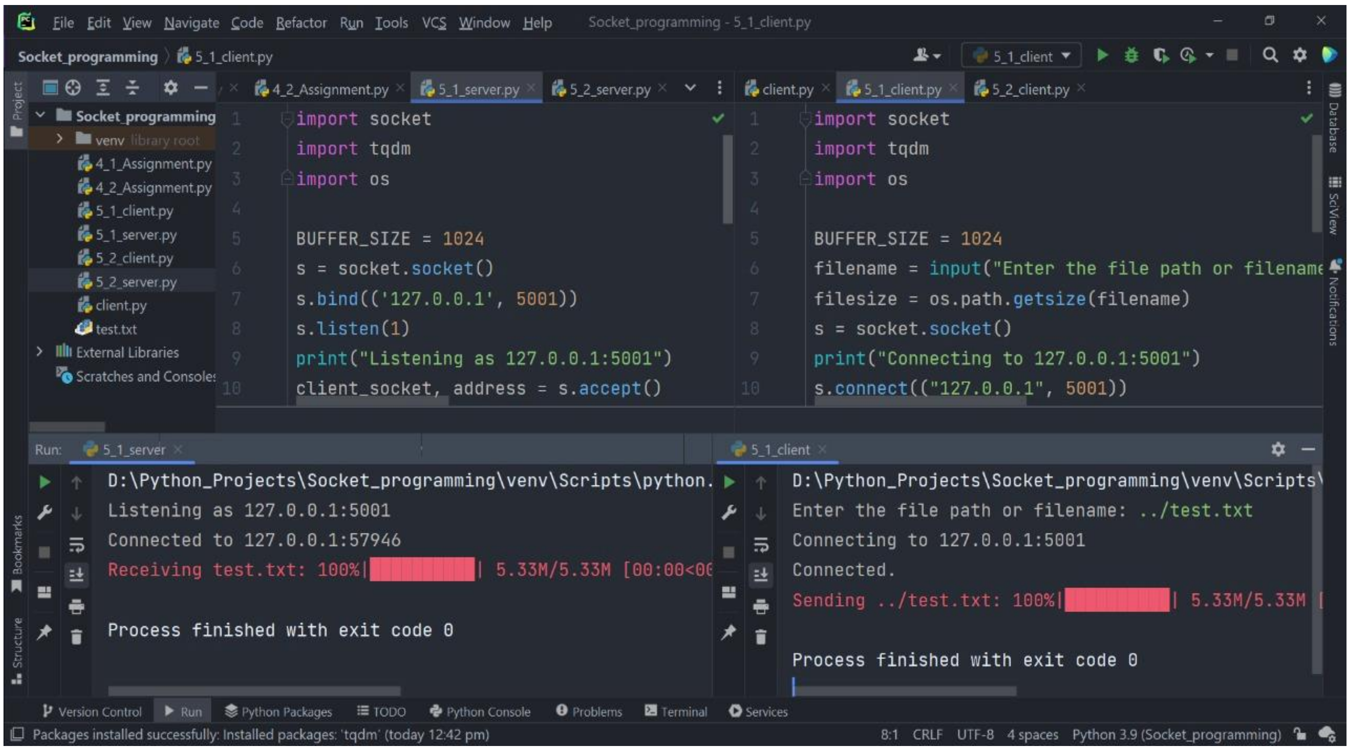
```
import socket
import tqdm
import os

BUFFER_SIZE = 1024
s = socket.socket()
s.bind(('127.0.0.1', 5001))
s.listen(1)
print("Listening as 127.0.0.1:5001")
client_socket, address = s.accept()
print(f"Connected to {address[0]}:{str(address[1])}")
filename, filesize =
client_socket.recv(BUFFER_SIZE).decode().split('||')
filename = os.path.basename(filename)
filesize = int(filesize)
progress = tqdm.tqdm(range(filesize), f"Receiving
{filename}", unit="B", unit_scale=True,
unit_divisor=1024)
with open(filename, "wb") as f:
    while True:
        bytes_read = client_socket.recv(BUFFER_SIZE)
        if not bytes_read:
            break
        f.write(bytes_read)
        progress.update(len(bytes_read))
client_socket.close()
s.close()
```

Client side code:

```
import socket
import tqdm
import os

BUFFER_SIZE = 1024
filename = input("Enter the file path or filename: ")
filesize = os.path.getsize(filename)
s = socket.socket()
print("Connecting to 127.0.0.1:5001")
s.connect(("127.0.0.1", 5001))
print("Connected.")
s.send(f"{filename}||{filesize}".encode())
```

```python
progress = tqdm.tqdm(range(filesize), f"Sending
{filename}", unit="B", unit_scale=True,
unit_divisor=1024)
with open(filename, "rb") as f:
    while True:
        bytes_read = f.read(BUFFER_SIZE)
        if not bytes_read:
            break
        s.sendall(bytes_read)
        progress.update(len(bytes_read))
s.close()
```

Output:



```python
import socket
import tqdm
import os

BUFFER_SIZE = 1024
s = socket.socket()
s.bind(('127.0.0.1', 5001))
s.listen(1)
print("Listening as 127.0.0.1:5001")
client_socket, address = s.accept()
```

```python
import socket
import tqdm
import os

BUFFER_SIZE = 1024
filename = input("Enter the file path or filename
filesize = os.path.getsize(filename)
s = socket.socket()
print("Connecting to 127.0.0.1:5001")
s.connect(("127.0.0.1", 5001))
```

```
D:\Python_Projects\Socket_programming\venv\Scripts\python.
Listening as 127.0.0.1:5001
Connected to 127.0.0.1:57946
Receiving test.txt: 100%|██████████| 5.33M/5.33M [00:00<0

Process finished with exit code 0
```

```
D:\Python_Projects\Socket_programming\venv\Scripts\
Enter the file path or filename: ../test.txt
Connecting to 127.0.0.1:5001
Connected.
Sending ../test.txt: 100%|██████████| 5.33M/5.33M |

Process finished with exit code 0
```

Packages installed successfully: Installed packages: 'tqdm' (today 12:42 pm)

2) Write a program to implement Remote Command Execution (RCE)

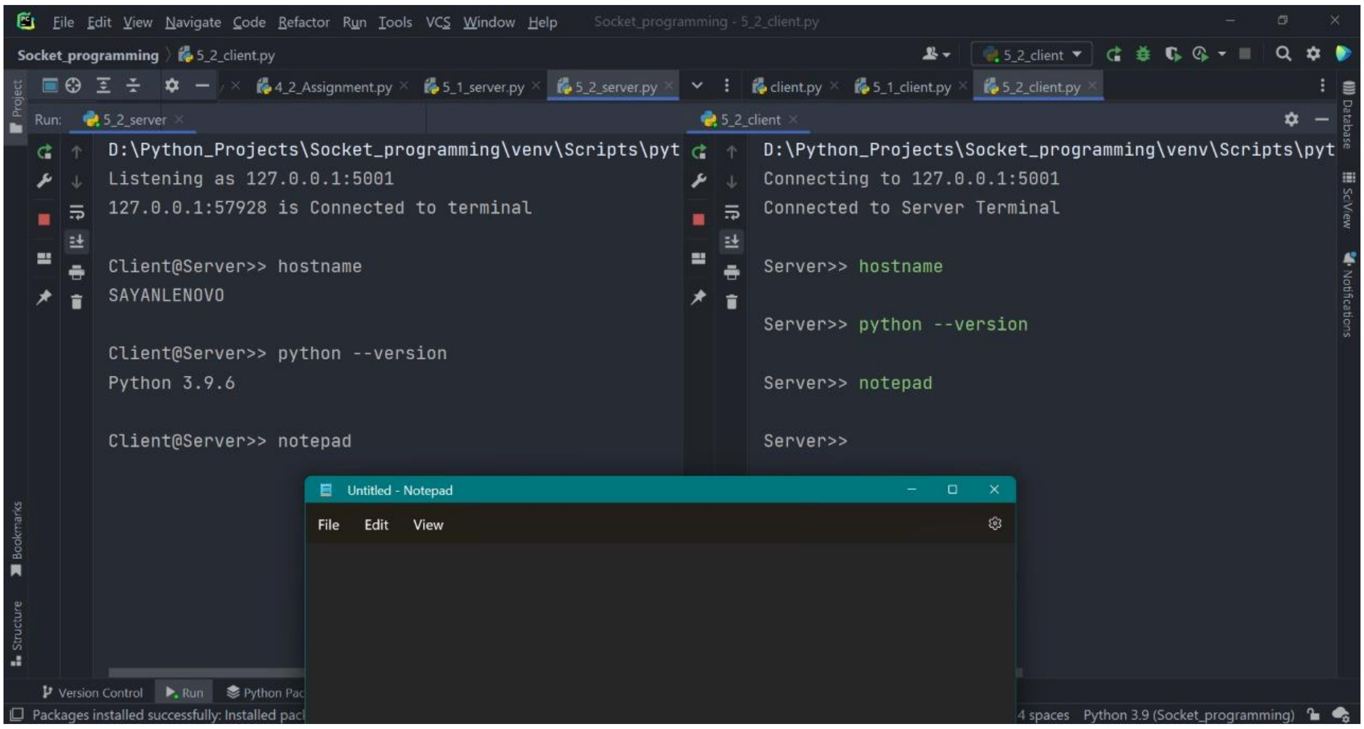
Server side code:

```python
import socket
import os

BUFFER_SIZE = 4096
s = socket.socket()
s.bind(('127.0.0.1', 5001))
s.listen(1)
print("Listening as 127.0.0.1:5001")
client_socket, address = s.accept()
print(f"{address[0]}:{str(address[1])} is Connected to
terminal")
while True:
    print("\nClient@Server>>", end=" ")
    command = client_socket.recv(BUFFER_SIZE).decode()
    print(command)
    if command == 'exit':
        break
    os.system(command)
print("Closing remote connection with client")
client_socket.close()
s.close()
```

Client side code:

```python
import socket

BUFFER_SIZE = 4096
s = socket.socket()
print("Connecting to 127.0.0.1:5001")
s.connect(("127.0.0.1", 5001))
print("Connected to Server Terminal")
while True:
    command = input("\nServer>> ")
    s.sendall(command.encode())
    if command == 'exit':
        break
print("Closing remote connection with Server")
s.close()
```

Output:


File Edit View Navigate Code Refactor Run Tools VCS Window Help
Socket_programming - 5_2_client.py
Socket_programming 5_2_client.py
4_2_Assignment.py
5_1_server.py
5_2_server.py
client.py
5_1_client.py
5_2_client.py
Run: 5_2_server
5_2_client
Untitled - Notepad
File Edit View
Version Control
Run
Python Pac
Packages installed successfully: Installed pacl
4 spaces Python 3.9 (Socket_programming)

```
D:\Python_Projects\Socket_programming\venv\Scripts\pyt
Listening as 127.0.0.1:5001
127.0.0.1:57928 is Connected to terminal

Client@Server>> hostname
SAYANLENOVO

Client@Server>> python --version
Python 3.9.6

Client@Server>> notepad
```

```
D:\Python_Projects\Socket_programming\venv\Scripts\pyt
Connecting to 127.0.0.1:5001
Connected to Server Terminal

Server>> hostname

Server>> python --version

Server>> notepad

Server>>
```