I. DOMAIN NAME SERVICE (DNS)

1. What is the IP address of your computer ? Is it IPv4 or IPv6 ?

My computer's IP address is 10.10.12.208
It's IPv4

2. What is the IP address of your smartphone ?

My smart phone's IP address is 10.10.16.163

3. What is the IP address of iutv.univ-paris13.fr ?

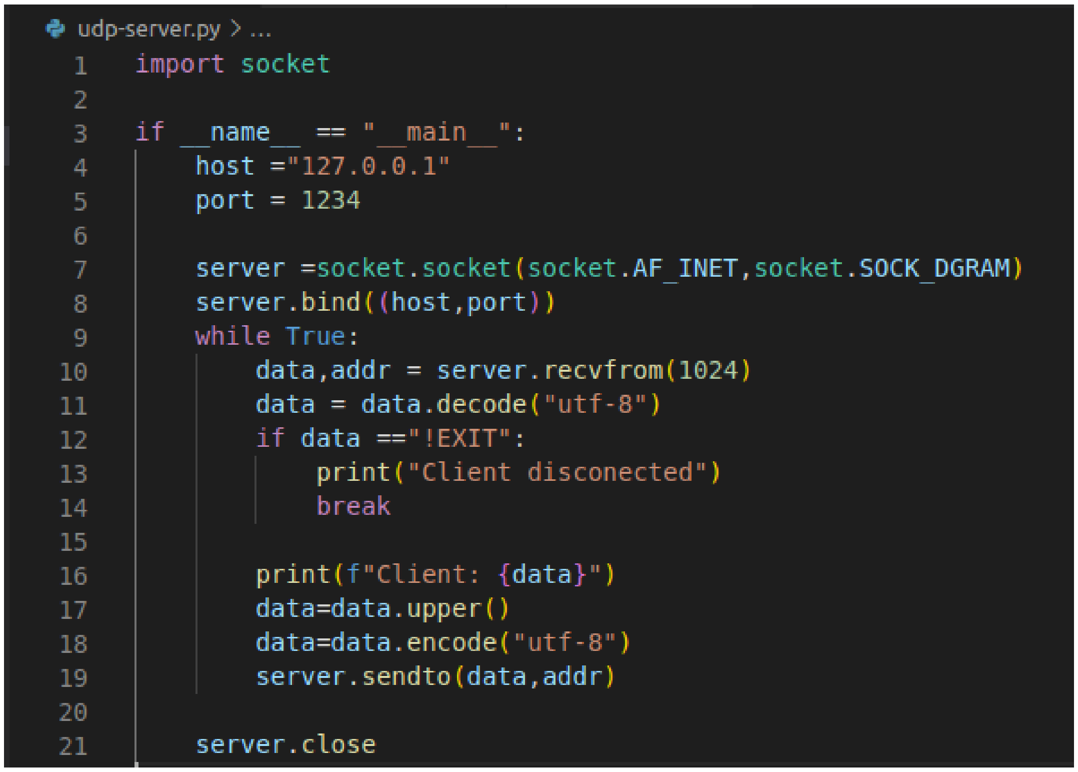
IP address of iutv.univ-paris13.fr is 81.194.43.211

4. What are the address of your DNS servers ?

The address of my DNS server is 192.168.0.1

II. UDP

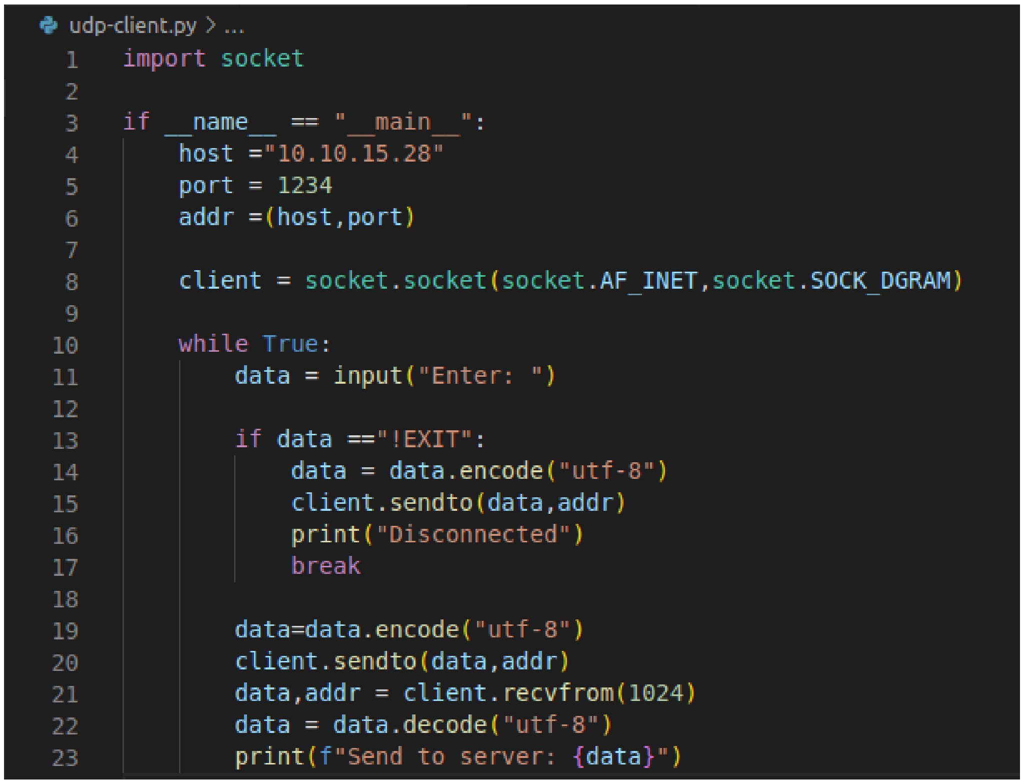
1. Write an UDP server, printing all received messages and never ending



udp-server.py > ...

```python
import socket

if __name__ == "__main__":
    host ="127.0.0.1"
    port = 1234

    server =socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
    server.bind((host,port))
    while True:
        data,addr = server.recvfrom(1024)
        data = data.decode("utf-8")
        if data =="!EXIT":
            print("Client disconected")
            break

        print(f"Client: {data}")
        data=data.upper()
        data=data.encode("utf-8")
        server.sendto(data,addr)

    server.close
```

2. **Write an UDP client, sending a message**

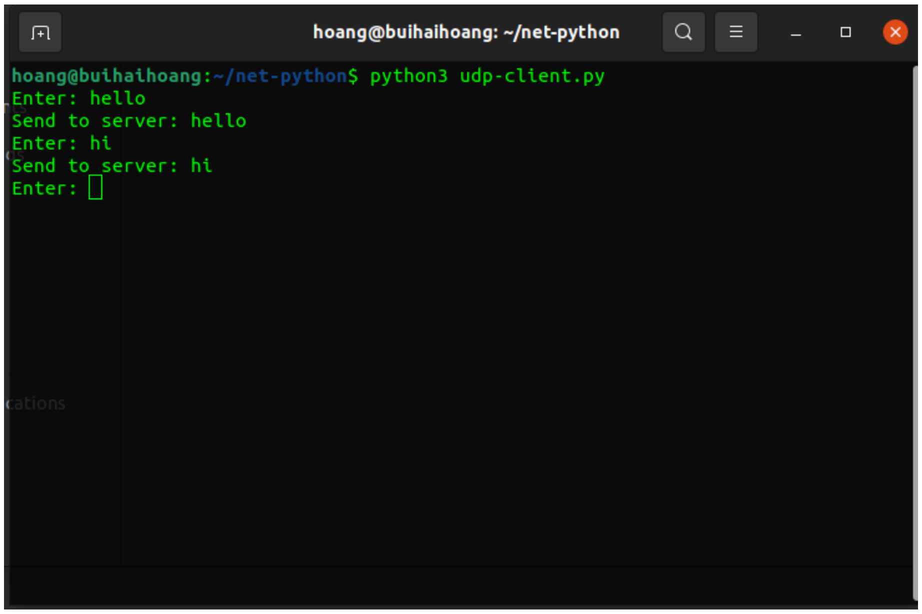


```
udp-client.py > ...
import socket

if __name__ == "__main__":
    host ="10.10.15.28"
    port = 1234
    addr =(host,port)

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

    while True:
        data = input("Enter: ")

        if data =="!EXIT":
            data = data.encode("utf-8")
            client.sendto(data,addr)
            print("Disconnected")
            break

        data=data.encode("utf-8")
        client.sendto(data,addr)
        data,addr = client.recvfrom(1024)
        data = data.decode("utf-8")
        print(f"Send to server: {data}")
```

3. **Using your udpclient, try to communicate with the udpserver of some other students**

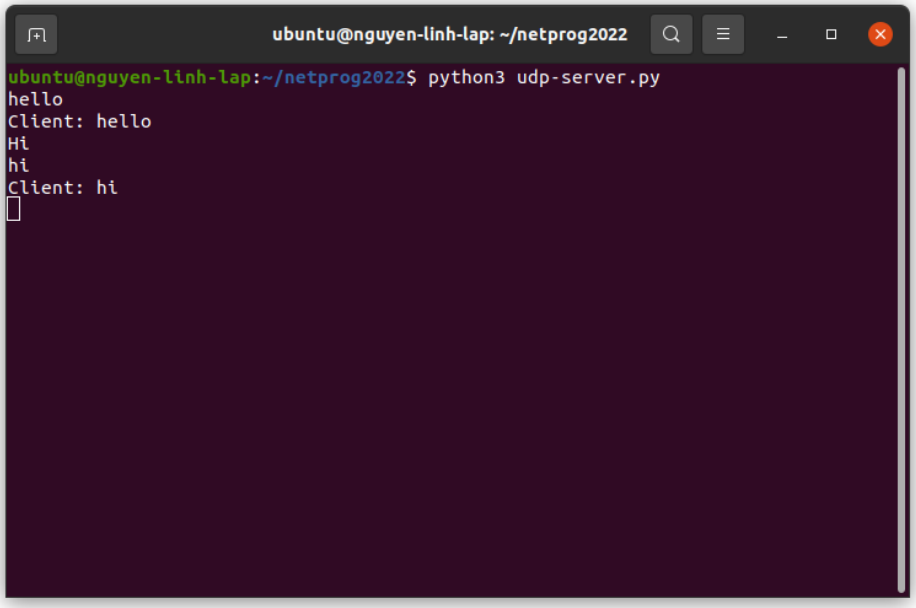
I am client, I connect with my friend (server) and sent to server some messages

- Client:



```
hoang@buihaihoang: ~/net-python
hoang@buihaihoang:~/net-python$ python3 udp-client.py
Enter: hello
Send to server: hello
Enter: hi
Send to server: hi
Enter: 
```

- Server:


ubuntu@nguyen-linh-lap: ~/netprog2022

```
ubuntu@nguyen-linh-lap:~/netprog2022$ python3 udp-server.py
hello
Client: hello
Hi
hi
Client: hi
```

4. **Use wireshark to show the network traffic from/to your computer**

- **What is the ARP traffic when you try to send a message to a new machine?**

ARP is 1 types of protocol to find MAC

- **Locate some UDP messages. What is the Ethernet frame size ? Compare it with the size of the text message sent.**

Frame size of Ethernet is 232 bytes on wire (1856 bits), 232 bytes captured (1856 bits)

- **Can wireshark read the text of the messages sent by udpclient ?**

Can't read the messages

III. TCP

1. **What are the types (classes) of cnx and addr ?**

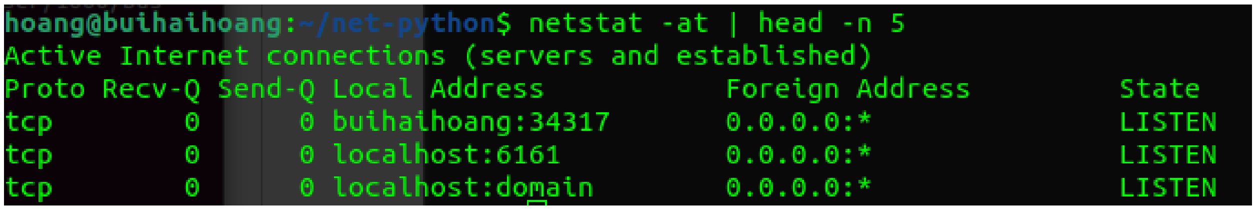
- cnx is known as a new socket object. Can be use to send and receive
- Addr bound to the socket on the other end

2. **Open 2 terminal windows on a linux system**

- **In the first one, launch a Python interpreter and start a TCP server on port 6161 (socket, bind, listen, accept) What are the “blocking” calls ?**

A blocking call doesn’t return until the request completes. blocking accept() call does not return to your program until a client connects to your socket program.

- **In the other terminal, use netstat and locate your server.**
- **The launch python and connect() to the server. What do you observe ?**



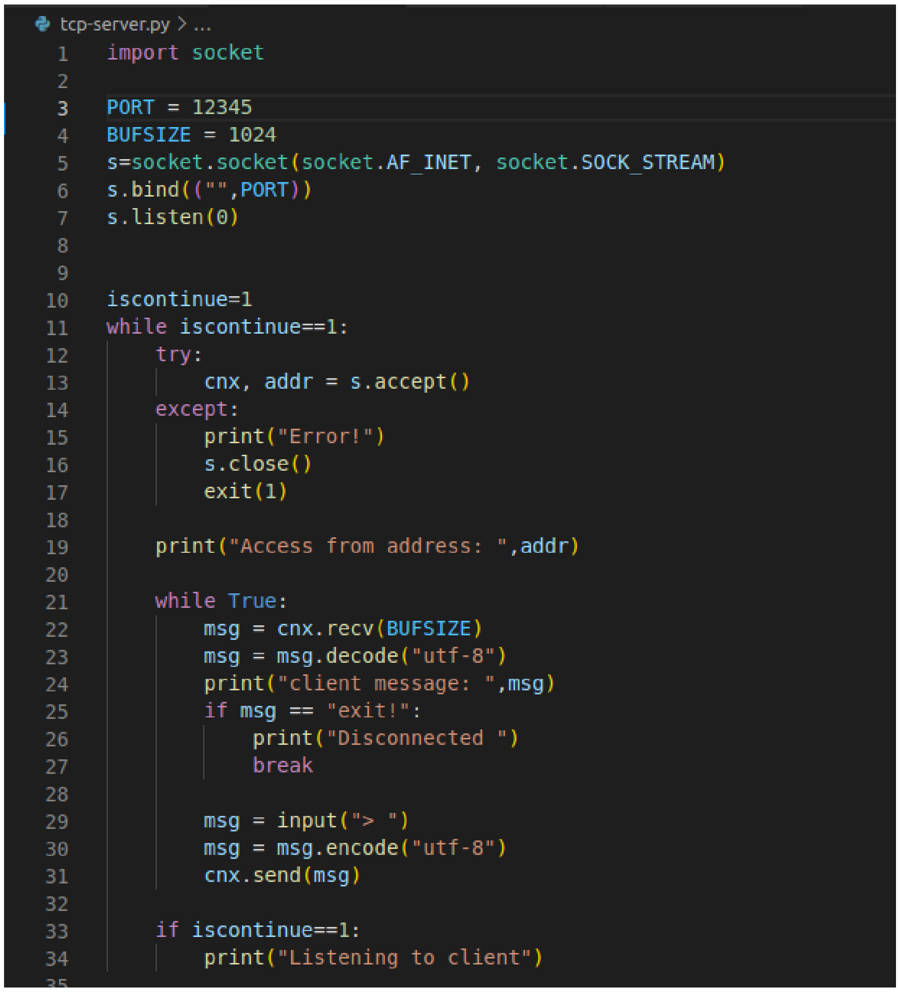
```
hoang@buihaihoang:~/net-python$ netstat -at | head -n 5
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 buihaihoang:34317       0.0.0.0:*               LISTEN
tcp        0      0 localhost:6161          0.0.0.0:*               LISTEN
tcp        0      0 localhost:domain        0.0.0.0:*               LISTEN
```

The client side will try to connect to all addresses, and send traffic to the first one connected successfully.

3. **How can we know when the connection is closed ?**

The method waits until the connection completes, or has an error on timeout, if the signal doesn't raise, it means the socket is blocking or has a timeout.
At this time, it will wait until connection completes instead of error.

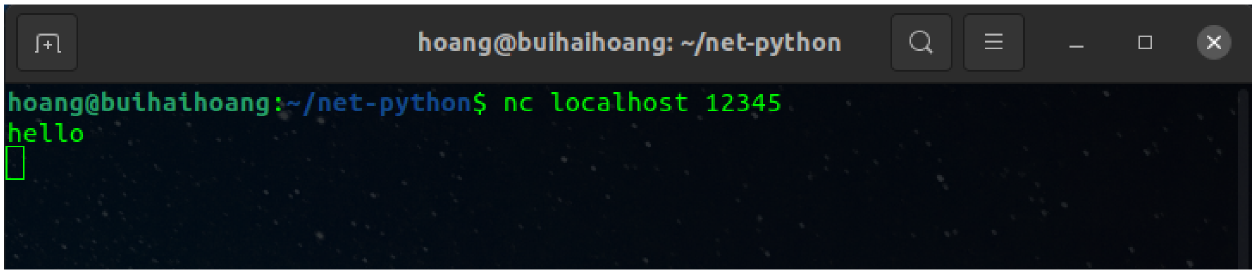
4. **Write a server, tcp-server.py, same usage as the UDP version. The server should : - print the received message - send back a text message to the client When the connection with the client is closed, the server should wait (accept) for another client.**



tcp-server.py > ...

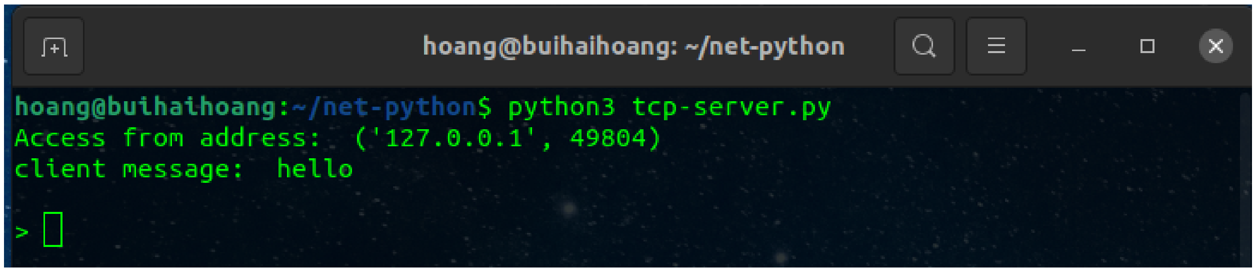
```python
import socket

PORT = 12345
BUFSIZE = 1024
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind(("",PORT))
s.listen(0)


iscontinue=1
while iscontinue==1:
    try:
        cnx, addr = s.accept()
    except:
        print("Error!")
        s.close()
        exit(1)

    print("Access from address: ",addr)

    while True:
        msg = cnx.recv(BUFSIZE)
        msg = msg.decode("utf-8")
        print("client message: ",msg)
        if msg == "exit!":
            print("Disconnected ")
            break

        msg = input("> ")
        msg = msg.encode("utf-8")
        cnx.send(msg)

    if iscontinue==1:
        print("Listening to client")
```

5. **Use the command nc to connect to your tcp-server**



hoang@buihaihoang: ~/net-python

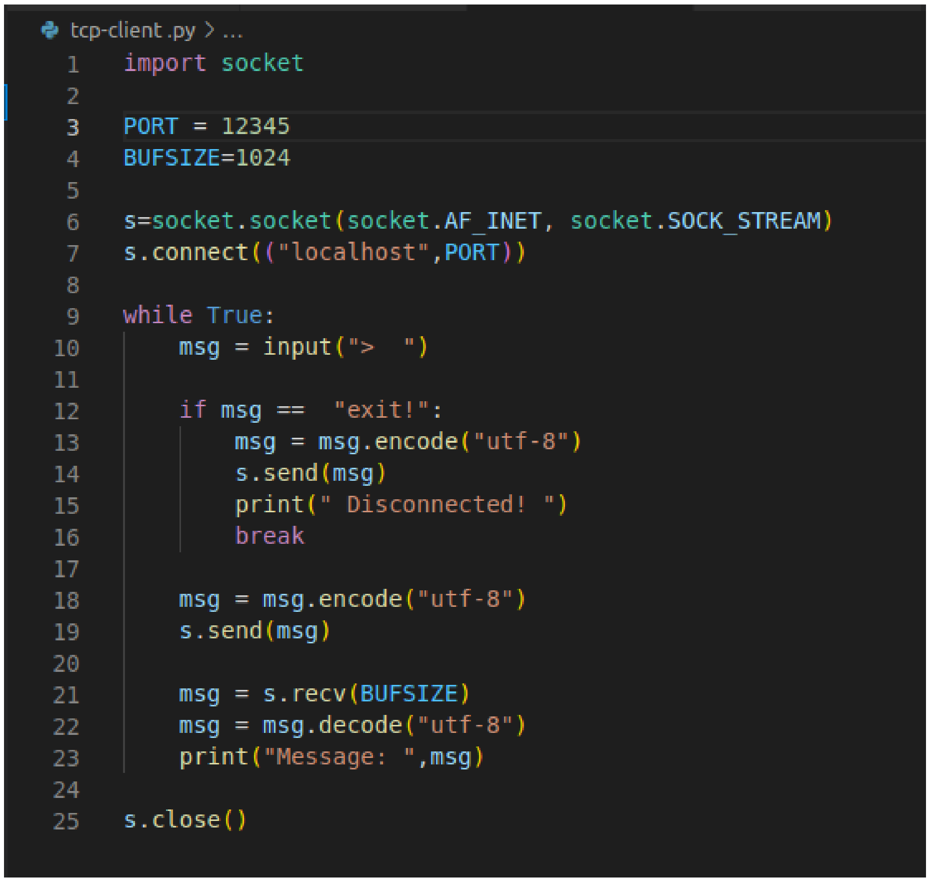
```
hoang@buihaihoang:~/net-python$ nc localhost 12345
hello
```



```
hoang@buihaihoang:~/net-python$ python3 tcp-server.py
Access from address:  ('127.0.0.1', 49804)
client message:  hello

> 
```

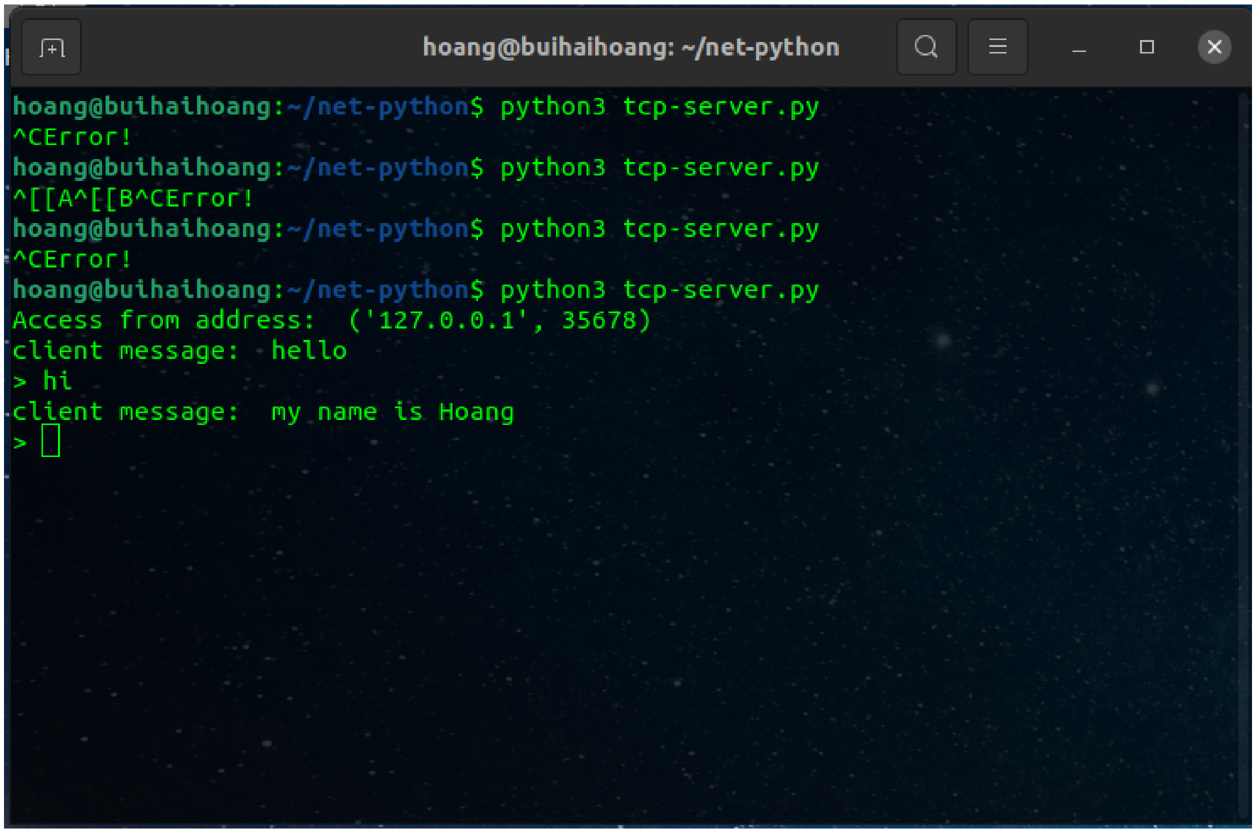
6. **Write tcp-client.py, same usage as UDP version.**
- **Send a message (string) to the server**
- **Read and display the response**
- **Close the connection**

Client:

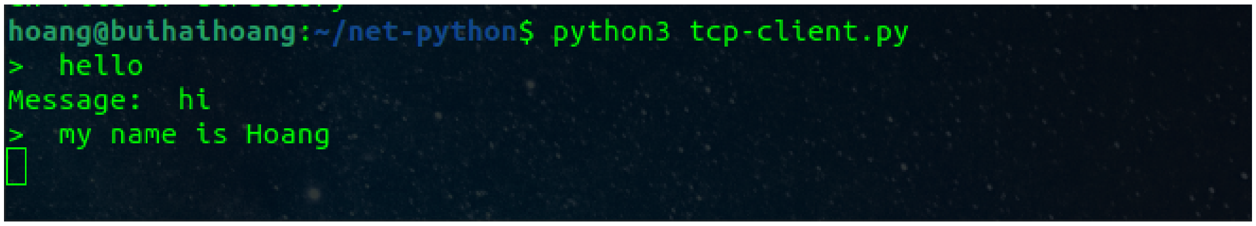


```python
import socket

PORT = 12345
BUFSIZE=1024

s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(("localhost",PORT))

while True:
    msg = input(">  ")

    if msg ==  "exit!":
        msg = msg.encode("utf-8")
        s.send(msg)
        print(" Disconnected! ")
        break

    msg = msg.encode("utf-8")
    s.send(msg)

    msg = s.recv(BUFSIZE)
    msg = msg.decode("utf-8")
    print("Message: ",msg)

s.close()
```

.



```
hoang@buihaihoang:~/net-python$ python3 tcp-server.py
^CError!
hoang@buihaihoang:~/net-python$ python3 tcp-server.py
^[[A^[[B^CError!
hoang@buihaihoang:~/net-python$ python3 tcp-server.py
^CError!
hoang@buihaihoang:~/net-python$ python3 tcp-server.py
Access from address:  ('127.0.0.1', 35678)
client message:  hello
> hi
client message:  my name is Hoang
> 
```



```
hoang@buihaihoang:~/net-python$ python3 tcp-client.py
>  hello
Message:  hi
>  my name is Hoang
```