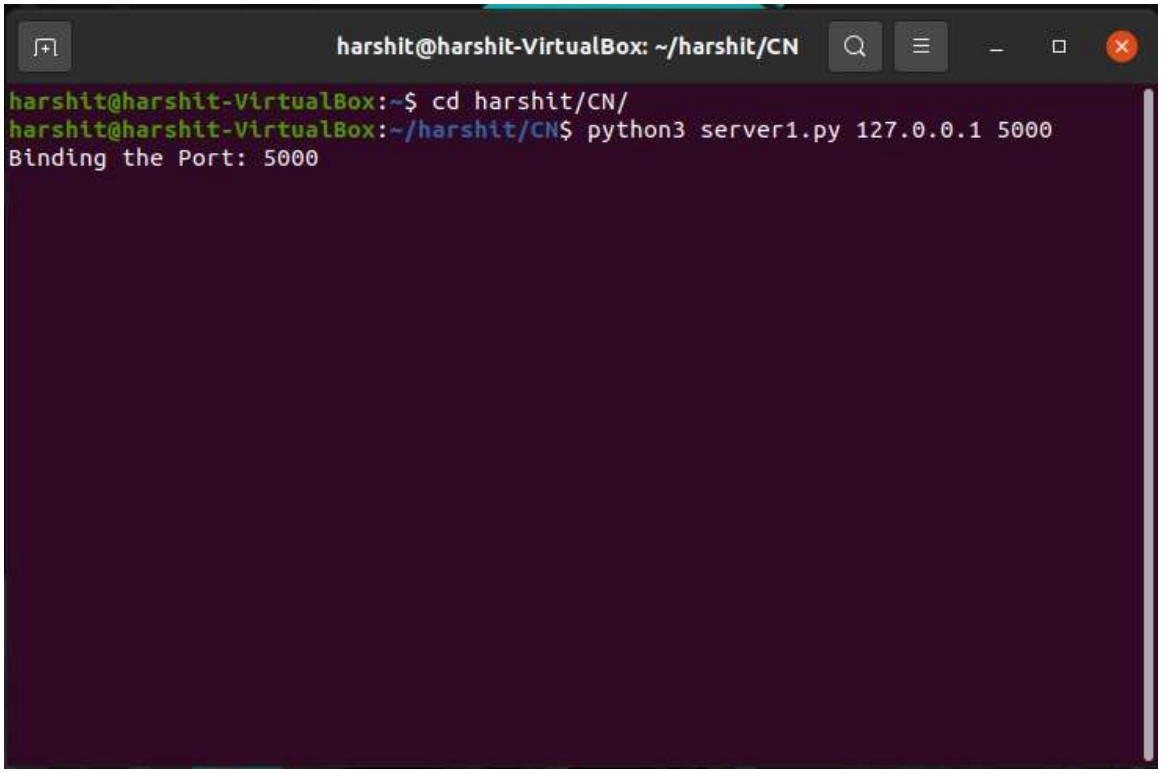
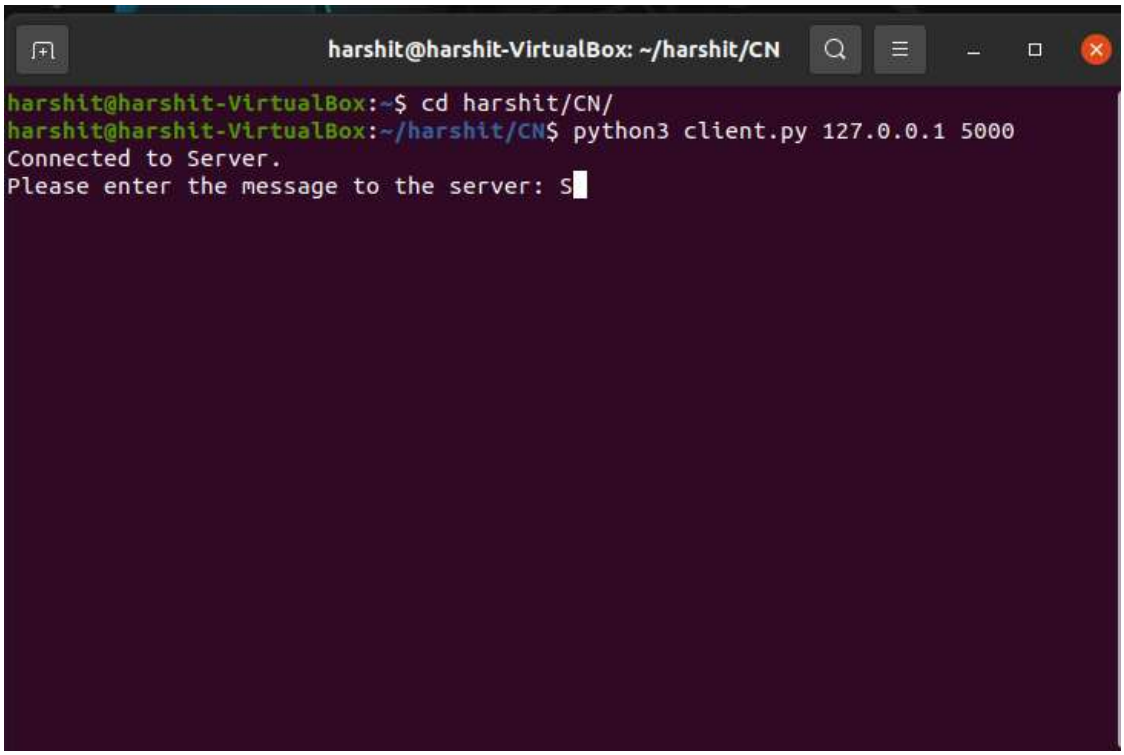
REPORT

Instructions to run the code locally:

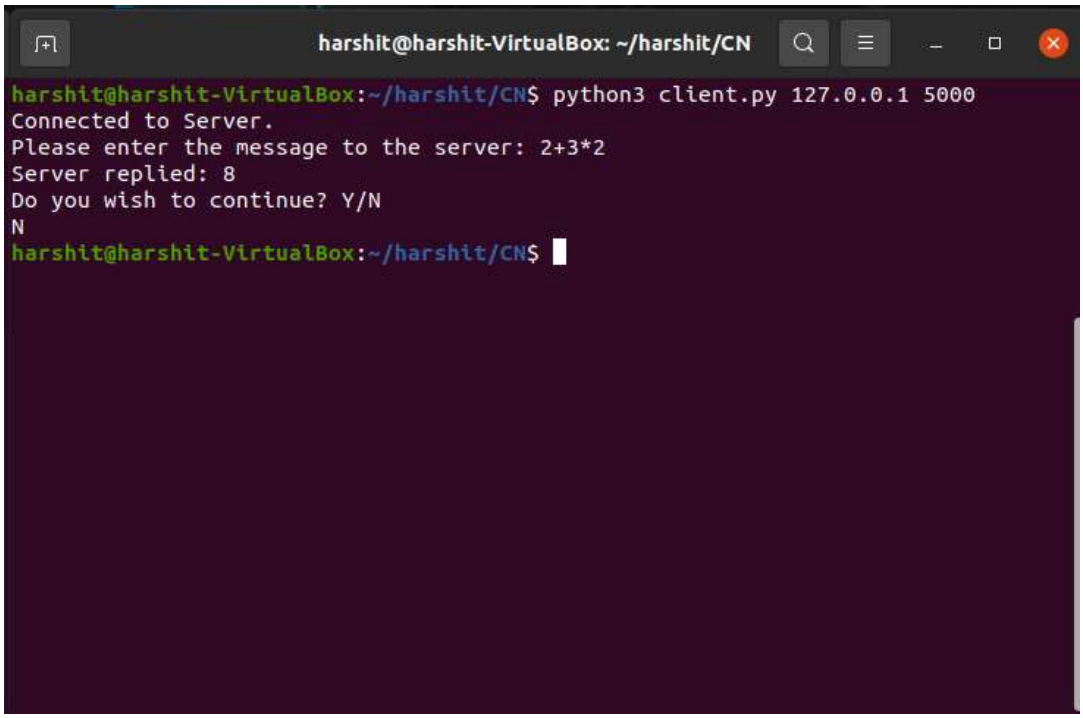
1. First, we will run the terminal and go to the directory where the server code is present.
2. Type in the following command in the terminal – python3 server1.py IP PORT.
3. We will use localhost i.e. 127.0.0.1 for IP and 5000 for PORT.



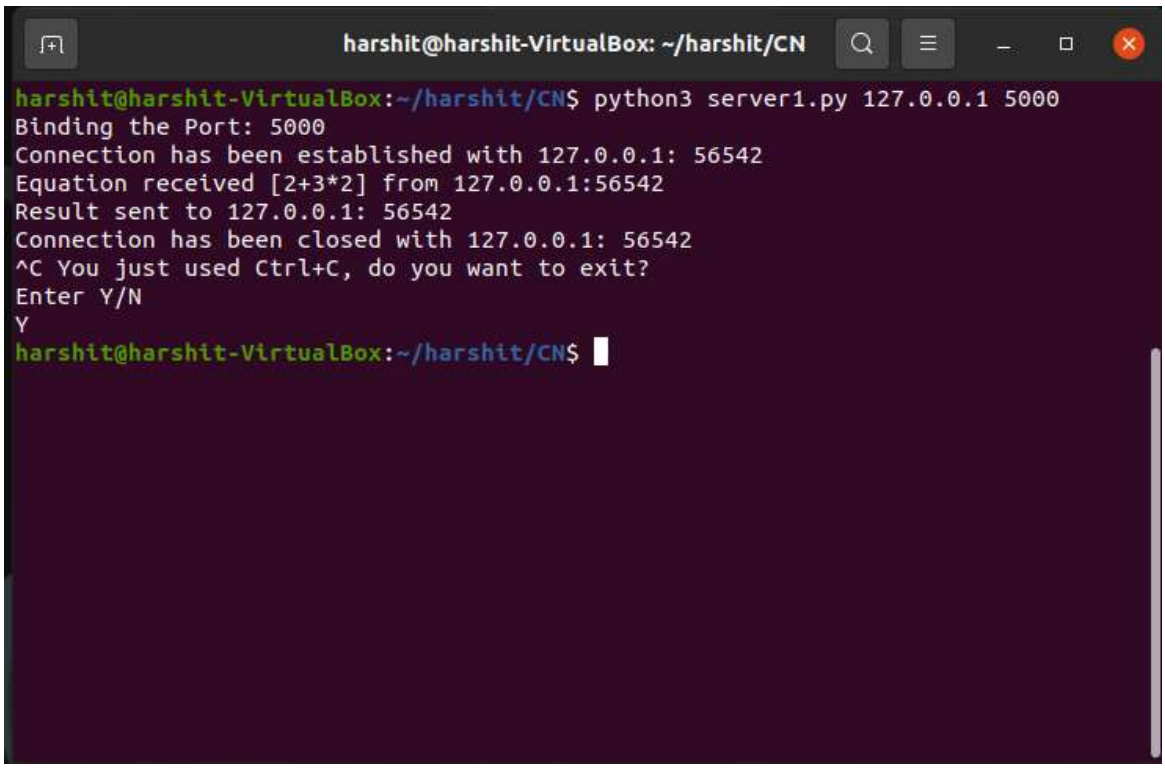
4. Open another terminal and open the directory where the client.py code is present.
5. Enter the following command – python3 client.py 127.0.0.1 5000 (the IP and port used by the server).



6. After this, the server-side terminal should prompt that "Connection has been established with 127.0.0.1: PORT"
7. Now the client can send requests to the server. Depending on the type of server single or multiple clients can be connected
8. In order to exit the client.py, after entering the query the program asks you whether you want to continue or not. Enter N to exit.



9. To exit server.py press ctrl+C and y get a prompt whether you want to exit or not. Press Y.



Instructions to run the code on different systems:

1. In step 3 of locally use ip of your own device and any free port and in step 5 use the ip of server device.

Screenshot of output on testcase 1:

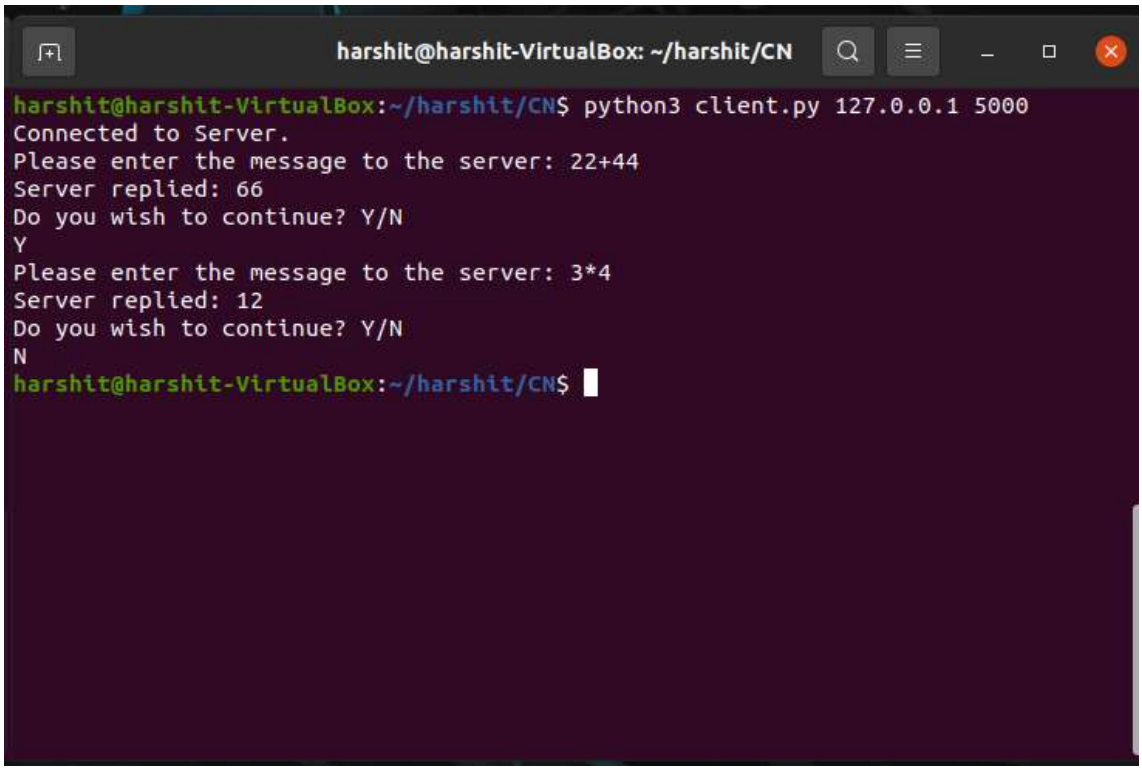
Testcase 1:

```
$ python client.py 127.0.0.1 5000
Connected to server
Please enter the message to the server: 22 + 44
Server replied: 66
Please enter the message to the server: 3 * 4
Server replied: 12
```

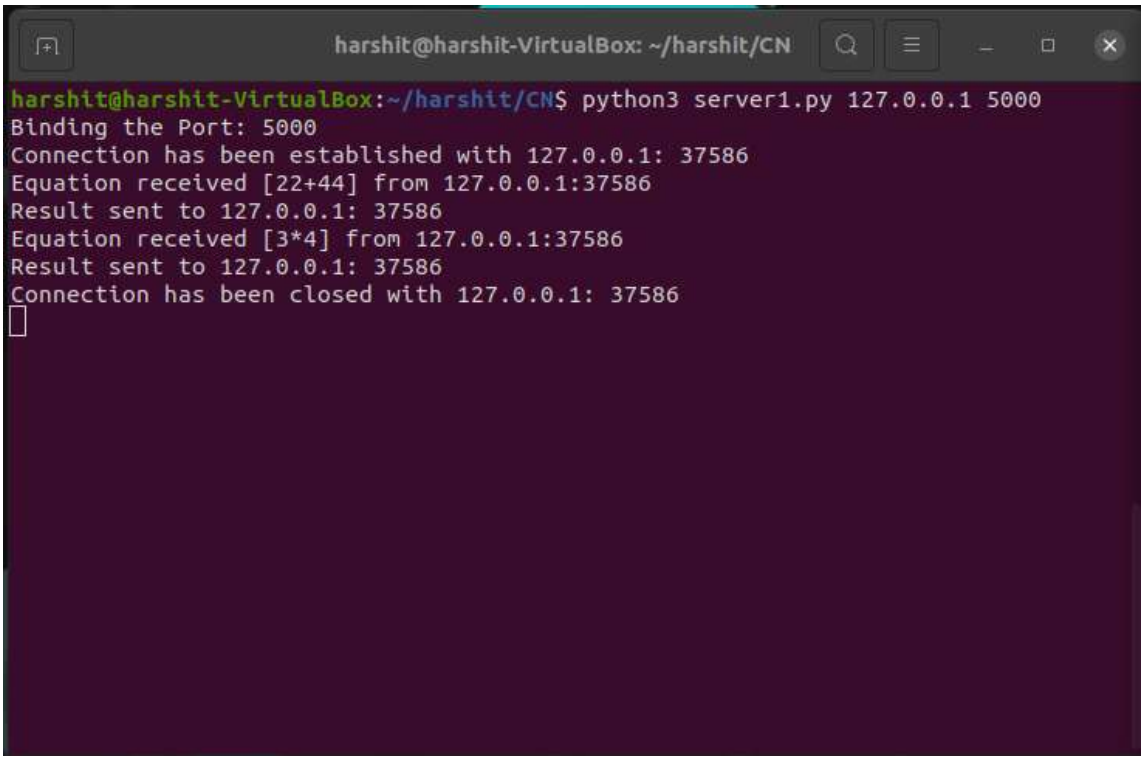
In this testcase we assume that only 1 client is allowed to be connected to the server .

Output:

Client:



Server:



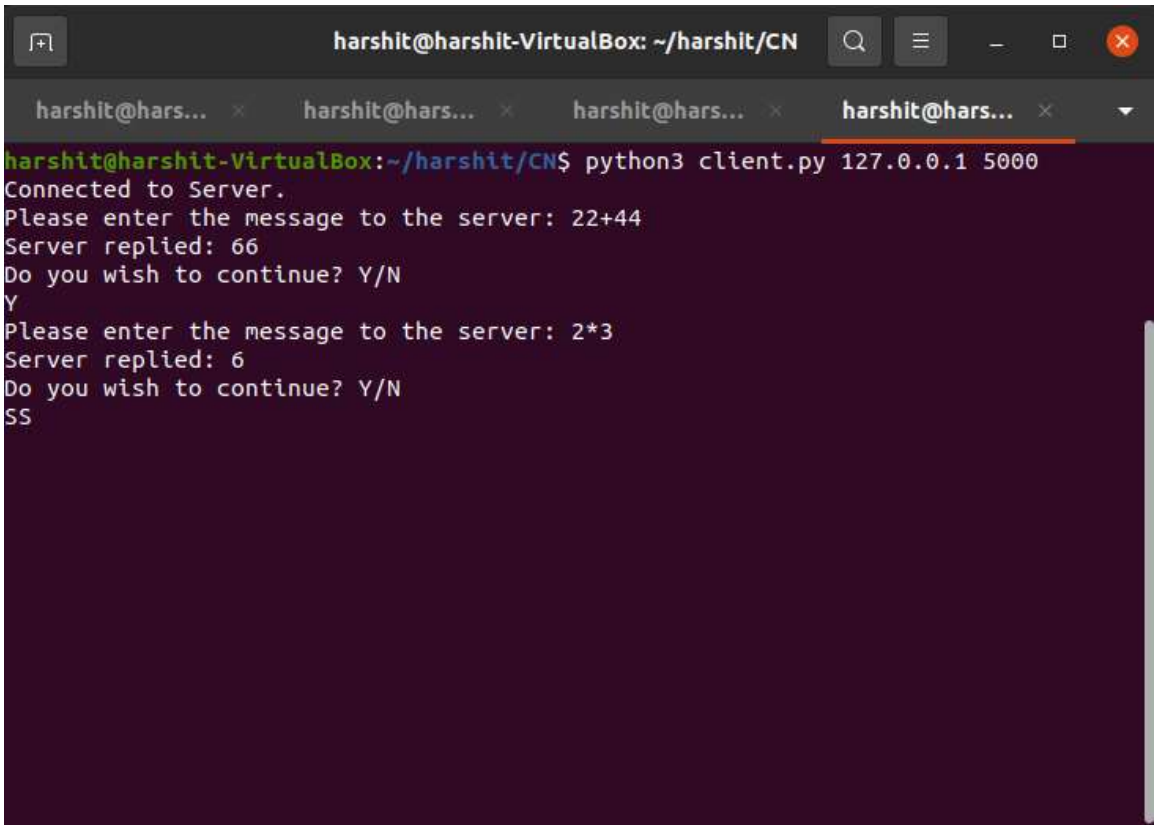
Testcase2:

```
$ python server1.py 127.0.0.1 5000
Connected with client socket number 4
Client socket 4 sent message: 22 + 44
Sending reply: 66
Client socket 4 sent message: 3 * 4
Sending reply: 12
```

In this testcase we assume that 4 clients are connected to the server and that these queries are from the 4th client, hence "socket 4".

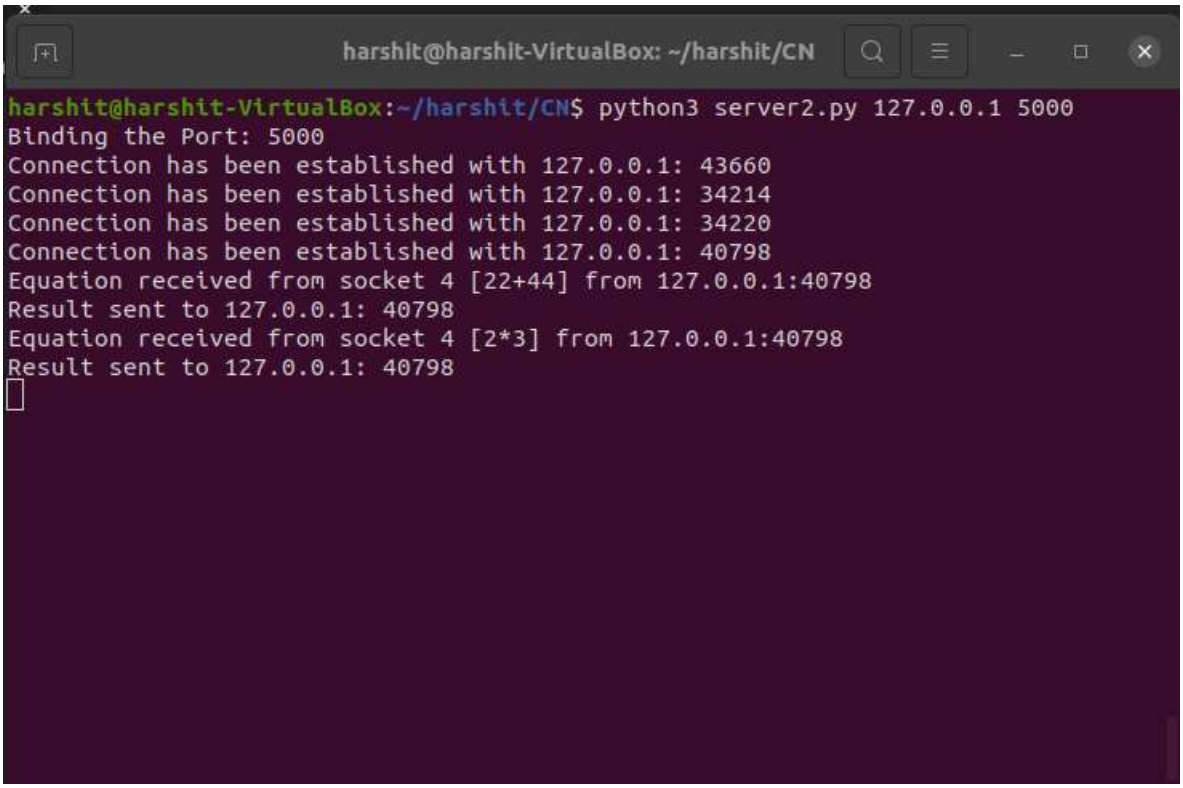
Output:

Client:



Here we can see that there 4 terminals open, each terminal has one client open and the visible terminal is the client no. 4.

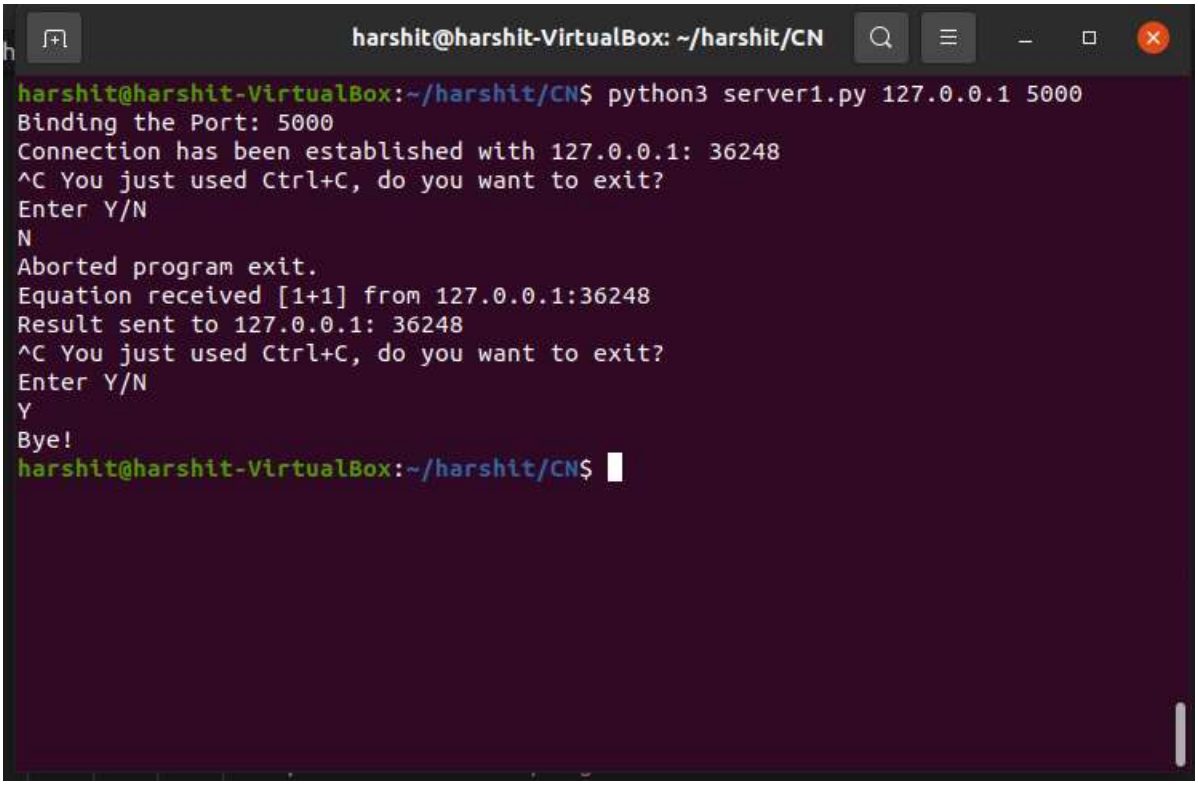
Server:



We can see that 4 connections are established and the queries obtained are from the 4th client and the server is mentioning this in its output.

Additional Features:

1. Once the user enters Ctrl+C to end the server program, instead of giving errors and terminating, we have done error handling and reconfirm with the user whether they want to exit or not and if yes, exit cleanly and if not then continue with the program execution.



Here, we can see that once we enter ctrl+c, the program asks the user whether they want to exit, if the user enters N, then exit is aborted and normal execution is continued.

When the user enters Y, the program exits with a Bye!