In this project, we build a simple client-server system where you use the client to chat with a "math" server.

There are two types of client-server system created :

1) Single Process Server: In this server program, "server1" will be a single process server that can handle only one client at a time. If a second client tries to chat with the server while one client's session is already in progress, the second client's socket operations should see an error.
2) Multiprocess Server: In this server program, "server2" will be a multi-process or multi-threaded server that will fork a process for every new client it receives. Multiple clients should be able to chat simultaneously with the server.

To run any of the above two client-server programs follow these steps:

1) Open two separate terminal windows. One will be used to run the server, and the other will run the client.
2) Using the cd command, navigate to the directory where your server.py and client.py files are located.
3)Run the server: In one terminal window, run the server program by executing the following command:

```
python server.py
```

The server will start running and will be ready to accept client connections.
4) Run the client: In the second terminal window, run the client program by executing the following command:

```
python client.py
```

Your client program will connect to the server, and you can start sending requests to perform math operations.

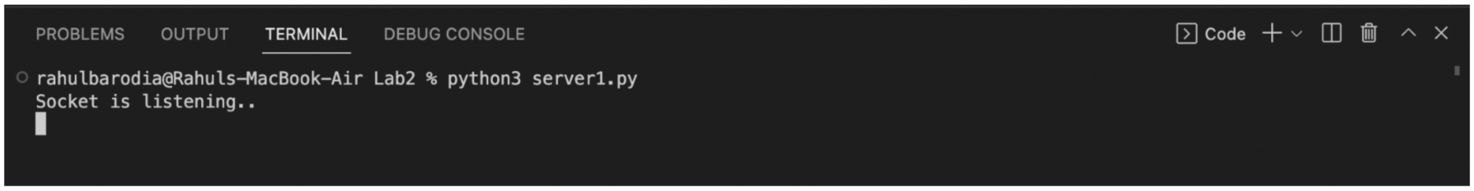
if you are using Python 3, you should use python3 instead of Python in the commands above.
Example python3 server.py

Alternatively, you can run both the client-server programs on VS code as follows :

Single Process Server :

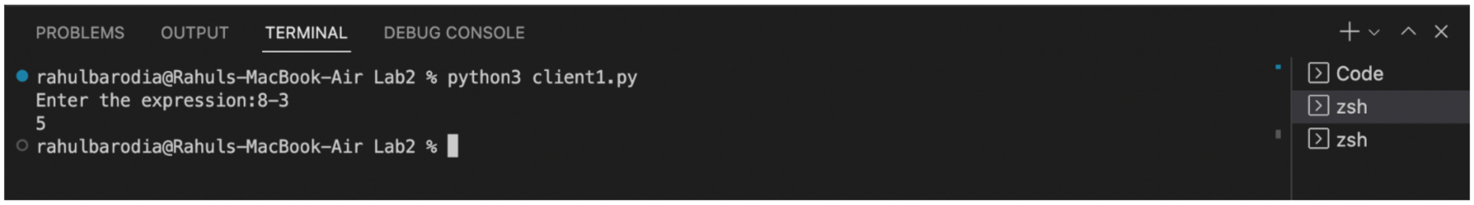
Output:

Server:

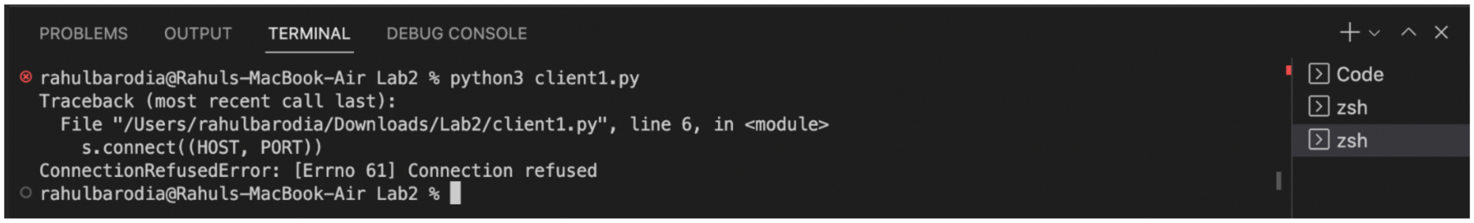


```
rahulbarodia@Rahuls-MacBook-Air Lab2 % python3 server1.py
Socket is listening..
```

Client:



```
rahulbarodia@Rahuls-MacBook-Air Lab2 % python3 client1.py
Enter the expression:8-3
5
rahulbarodia@Rahuls-MacBook-Air Lab2 %
```



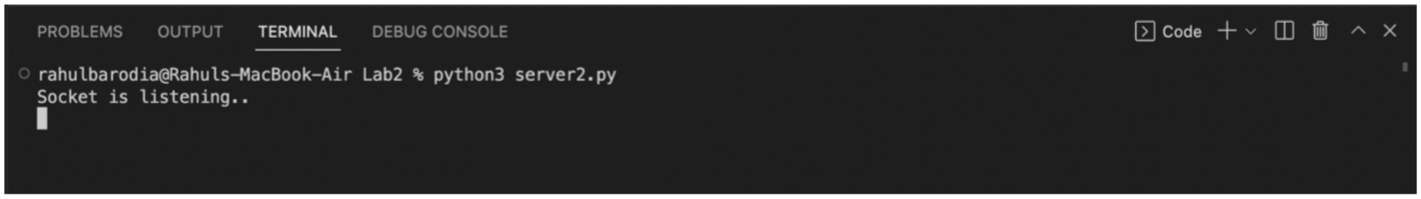
```
rahulbarodia@Rahuls-MacBook-Air Lab2 % python3 client1.py
Traceback (most recent call last):
  File "/Users/rahulbarodia/Downloads/Lab2/client1.py", line 6, in <module>
    s.connect((HOST, PORT))
ConnectionRefusedError: [Errno 61] Connection refused
rahulbarodia@Rahuls-MacBook-Air Lab2 %
```

As we can see from the above images that when a 2nd client tries to access the server, the connection is refused. Thus server1 can handle only a single client at a time.

Multiprocess Server :

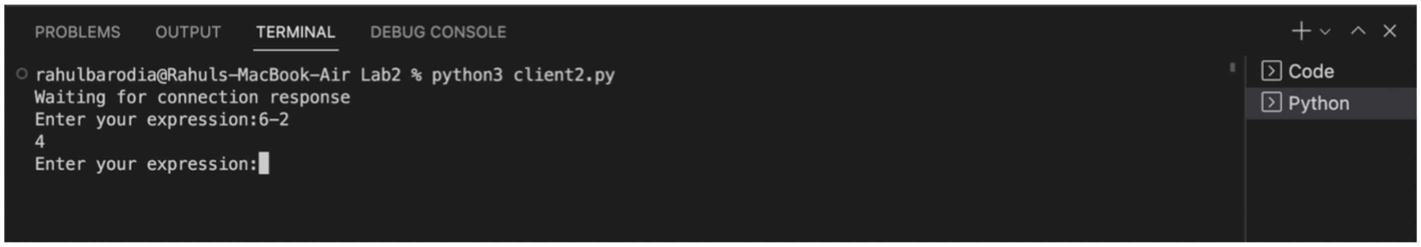
Output:

Server:

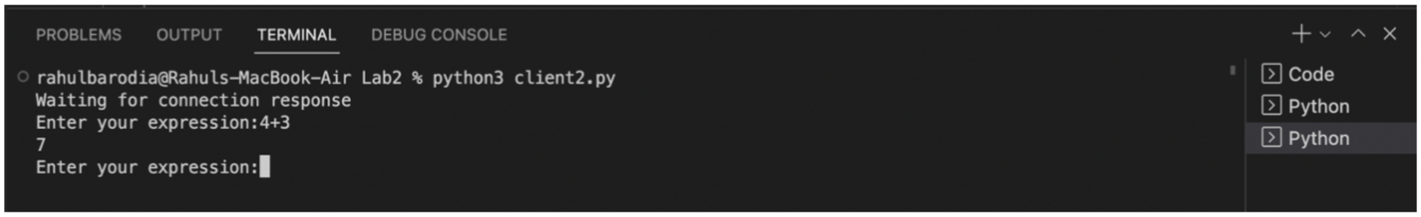


```
rahulbarodia@Rahuls-MacBook-Air Lab2 % python3 server2.py
Socket is listening..
```

Client:



```
rahulbarodia@Rahuls-MacBook-Air Lab2 % python3 client2.py
Waiting for connection response
Enter your expression:6-2
4
Enter your expression:
```



```
rahulbarodia@Rahuls-MacBook-Air Lab2 % python3 client2.py
Waiting for connection response
Enter your expression:4+3
7
Enter your expression:
```

As we can see from the above images, server2 allows multiple clients to access at the same time. The above example shows two clients accessing server2 and getting the correct response.