ASSIGNMENT-2 Client Server Socket Programming

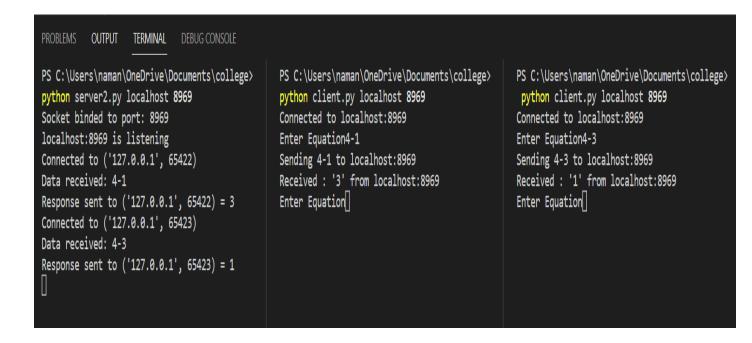
- Part1: Server handling single connection at a time
- Part2: Server handling multiple connections simultaneously
- The sockets must use TCP and IPV4 protocols.
- All the messages of connection details and information passed should be displayed on the command line with respect to servers and clients.
- Python3 must be installed.

- How To Run:
- o Each of the server files can be run using:
 - -> python serverx.py "host" "port"
- \circ Client files can be run using:
 - -> python client.py "host" "port"

1. Your server program "server1.py" 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 some other client's session is already in progress, the second client's socket operations should see an error. After the first client closes the connection, the server should then accept connections from the other client. server1.py:

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE		
<pre>PS C:\Users\naman\OneDrive\Documents\college> python server1.py localhost 8998 Socket binded to port: 8998 Listening at localhost:8998 Connected to ('127.0.0.1', 50637) Data received: 6<3 Response sent to ('127.0.0.1', 50637) = False [</pre>	PS C:\Users\naman\OneDrive\Documents\college> python client.py localhost 8998 Connected to localhost:8998 Enter Equation 6<3 Sending 6<3 to localhost:8998 Received: 'False' from localhost:8998 Enter Equation	PS C:\Users\naman\OneDrive\Documents\college > python client.py localhost 8998 Connected to localhost:8998 Enter Equation7-9 Sending 7-9 to localhost:8998 timed out PS C:\Users\naman\OneDrive\Documents\college > [

2. Your server program "server2.py" will be a multi-threaded server that will create a new thread for every new client request it receives. Multiple clients should be able to simultaneously chat with the server server2.py:



Each connection spawns its own thread and engages the client. Connection is closed when the client closes the connection.

3. Your server program "server3.py" will be a single process server that uses the "select" method to handle multiple clients concurrently server3.py:

PROBLEMS OUTPUT **TERMINAL** DEBUG CONSOLE PS C:\Users\naman\OneDrive\Documents\college> PS C:\Users\naman\OneDrive\Documents\college> PS C:\Users\naman\OneDrive\Documents\college> python server3.py localhost 8972 python client.py localhost 8972 python client.py localhost 8972 Socket binded to port: 8972 Connected to localhost:8972 Connected to localhost:8972 Listening at localhost:8972 Enter Equation4*3 Enter Equation5-3 Received request from ('127.0.0.1', 65434) Sending 4*3 to localhost:8972 Sending 5-3 to localhost:8972 Data received: 4*3 Received: '12' from localhost:8972 Received: '2' from localhost:8972 Enter Equation 4+4 Response sent = 12 Enter Equation2/1 Received request from ('127.0.0.1', 65435) Sending 4+4 to localhost:8972 Sending 2/1 to localhost:8972 Data received: 5-3 Received: '8' from localhost:8972 Received: '2.0' from localhost:8972 Enter Equation Enter Equation Response sent = 2 Data received: 4+4 Response sent = 8 Data received: 2/1 Response sent = 2.0

4. Your server program "server4.py" will be an echo server (that replies the same message to the client that was received from the same client); it will be a single process server that uses the "select" method to handle multiple clients concurrently server4.py:

PROBLEMS OUTPUT **TERMINAL** DEBUG CONSOLE PS C:\Users\naman\OneDrive\Documents\college> PS C:\Users\naman\OneDrive\Documents\college> PS C:\Users\naman\OneDrive\Documents\college> python server4.py localhost 8988 python client.py localhost 8988 python client.py localhost 8988 Connected to localhost:8988 Socket binded to port: 8988 Connected to localhost:8988 Listening at localhost:8988 Enter Equation 3+5 Enter Equation 3*7 Sending 3+5 to localhost:8988 Sending 3*7 to localhost:8988 Received request from ('127.0.0.1', 65445) Received request from ('127.0.0.1', 65446) Received: '3+5' from localhost:8988 Received: '3*7' from localhost:8988 Enter Equation Enter Equation Data received: 3+5 Response sent = 3+5 Data received: 3*7 Response sent = 3*7

Edge cases:

Keyboard interrupt ctrl+c

Invalid argument

```
PS C:\Users\naman\OneDrive\Documents\college>
ppython server1.py localhost 8799
Socket binded to port: 8799
Listening at localhost:8799

| C:\Users\naman\OneDrive\Documents\college
| python client.py localhost 8799 c--
Invalid Arguments: python client.py 'host' '
port' 'equation'
| PS C:\Users\naman\OneDrive\Documents\college
| > | |
```

Connecting to an unopened port

```
PS C:\Users\naman\OneDrive\Documents\college>

python client.py localhost 8977

Port not connected

PS C:\Users\naman\OneDrive\Documents\college>
```

Connecting to an already opened port

```
PS C:\Users\naman\OneDrive\Documents\college>
python server1.py localhost 8998
Socket binded to port: 8998
Listening at localhost:8998

[]

PS C:\Users\naman\OneDrive\Documents\college>
python server1.py localhost 8998
Port 8998 is already in use
PS C:\Users\naman\OneDrive\Documents\college>
[]
```

THANK YOU, Naman Jain BT20CSE087