OS lab 3

Ridhima Kohli
B19CSE071

Files :

server1.c - attends one client at a time and blocks other clients trying to connect

server2.c - uses forks to attend multiple clients at a time

server3.c - uses threads to attend multiple clients at a time

client.c - client program

How to run :

- gcc server1.c -o s1 **followed by** ./s1 5555
- gcc server2.c -o s2 **followed by** ./s2
- gcc server3.c -o s3 -lpthread **followed by** ./s3
- gcc client.c -o c **followed by** ./c 127.0.0.1 5555

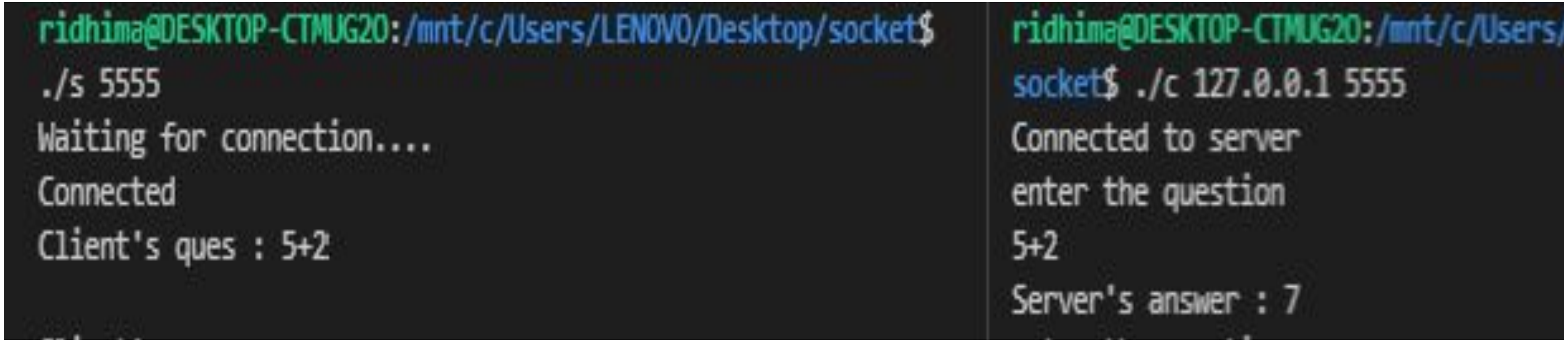
Make sure to specify the port number in server1
And address and port in client

For server3 since the pthread.h library has been used , so while compiling its necessary to write -lpthread as shown

To stop press ctrl+c

Simple 1 to 1 Server Client Communication - functions used for basic sockets :

#include <sys/socket.h> → Library used for implementation
sockaddr_in struct used for storing server and client informations
socket() for creating a socket
bind() for binding the socket to address and port
listen() to make client listen to the socket



```
ridhima@DESKTOP-CTMUG2O:/mnt/c/Users/LENOVO/Desktop/socket$
./s 5555
Waiting for connection....
Connected
Client's ques : 5+2
```

```
ridhima@DESKTOP-CTMUG2O:/mnt/c/Users/
socket$ ./c 127.0.0.1 5555
Connected to server
enter the question
5+2
Server's answer : 7
```

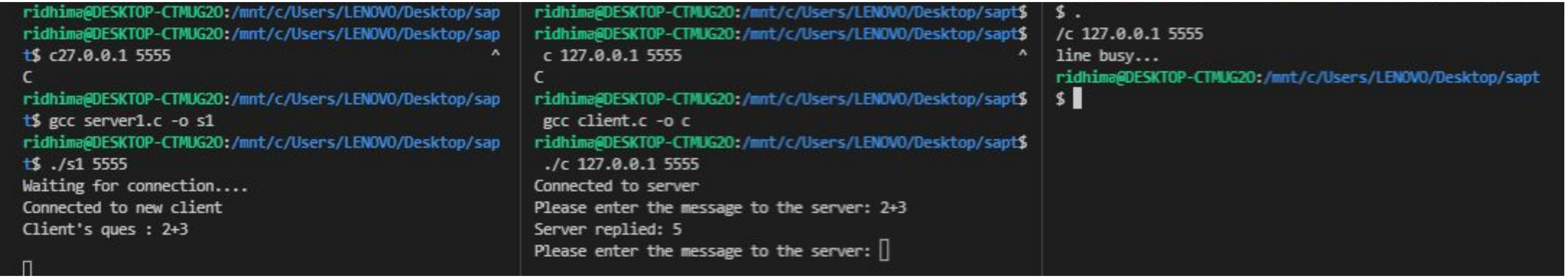
Strchr- used for identifying operator
Strtok - used for getting integer tokens

The calculated answer is then parsed into buffer’s data type by sprintf

Server 1 Outputs

The other client can’t connect as server is busy

For this feature , we close the listen fd as soon as a client is connected to let other clients know that server is busy and hence they cannot connect



```
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sap
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sap
t$ c27.0.0.1 5555                                     ^
C
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sap
t$ gcc server1.c -o s1
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sap
t$ ./s1 5555
Waiting for connection....
Connected to new client
Client's ques : 2+3
```

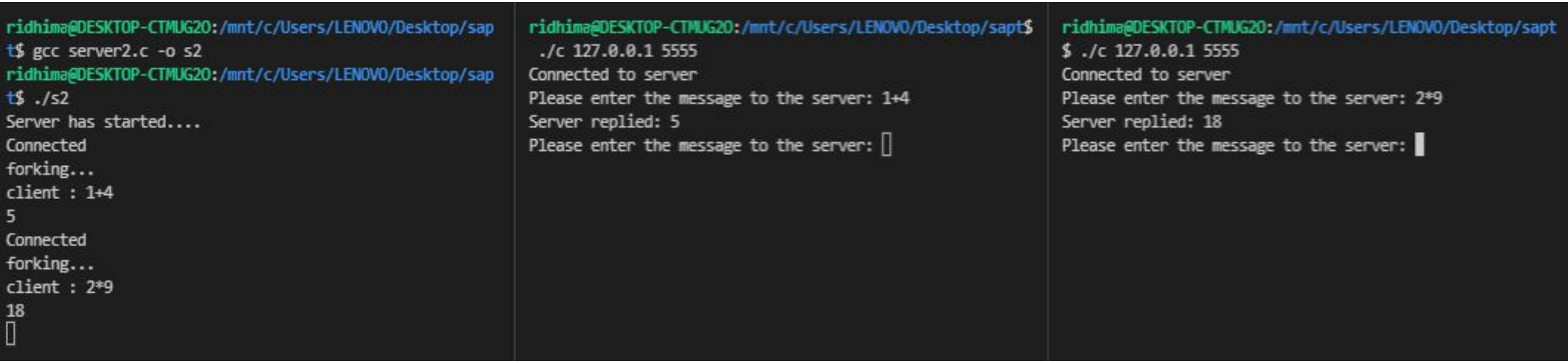
```
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sapt$
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sapt$
 c 127.0.0.1 5555                                      ^
C
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sapt$
 gcc client.c -o c
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sapt$
 ./c 127.0.0.1 5555
Connected to server
Please enter the message to the server: 2+3
Server replied: 5
Please enter the message to the server:
```

```
$ .
/c 127.0.0.1 5555
line busy...
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sapt
$
```

Server 2 Outputs

Multi Client service - using fork

Here we distribute the tasks of setting up connection and request handling among parent and child processes respectively



```
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sap
t$ gcc server2.c -o s2
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sap
t$ ./s2
Server has started....
Connected
forking...
client : 1+4
5
Connected
forking...
client : 2*9
18
```

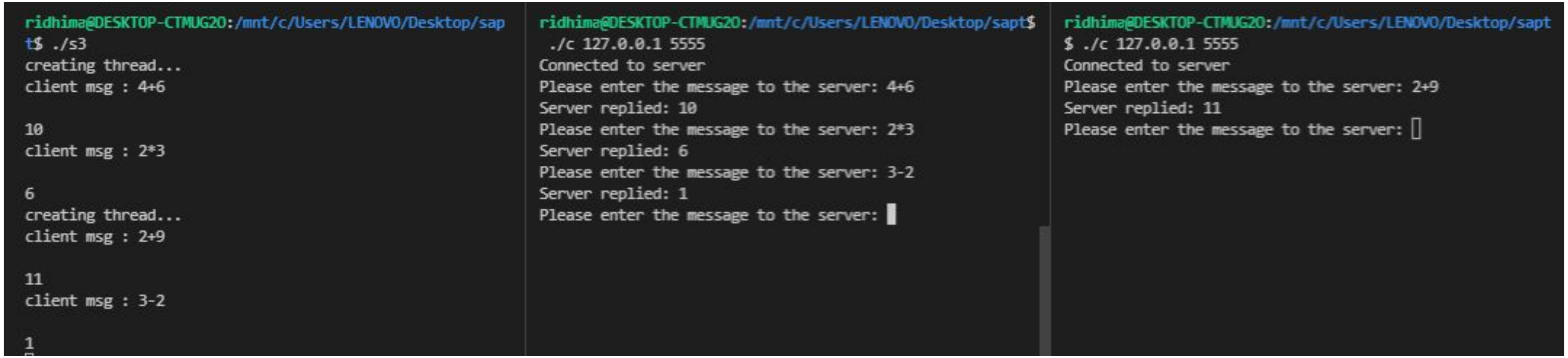
```
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sapt$
 ./c 127.0.0.1 5555
Connected to server
Please enter the message to the server: 1+4
Server replied: 5
Please enter the message to the server:
```

```
ridhima@DESKTOP-CTMUG20:/mnt/c/Users/LENOVO/Desktop/sapt
$ ./c 127.0.0.1 5555
Connected to server
Please enter the message to the server: 2*9
Server replied: 18
Please enter the message to the server:
```

Server 3 Outputs

Multi Client service using threads

Here we create a thread for handling of clients in different threads



```
ridhima@DESKTOP-CTMUG2O:/mnt/c/Users/LENOVO/Desktop/sap
t$ ./s3
creating thread...
client msg : 4+6

10
client msg : 2*3

6
creating thread...
client msg : 2+9

11
client msg : 3-2

1
```

```
ridhima@DESKTOP-CTMUG2O:/mnt/c/Users/LENOVO/Desktop/sapt$
 ./c 127.0.0.1 5555
Connected to server
Please enter the message to the server: 4+6
Server replied: 10
Please enter the message to the server: 2*3
Server replied: 6
Please enter the message to the server: 3-2
Server replied: 1
Please enter the message to the server:
```

```
ridhima@DESKTOP-CTMUG2O:/mnt/c/Users/LENOVO/Desktop/sapt
$ ./c 127.0.0.1 5555
Connected to server
Please enter the message to the server: 2+9
Server replied: 11
Please enter the message to the server:
```

Performance Comparison

Criteria	Server 1	Server 2 (multiple processes)	Server 3 (multiple threads)
Clients attended	Single client at a time	Multiple at a time	Multiple at a time
Comparison	Server 1 < server 2 = server 3		
Time taken for response	Almost equal for all since testing has been done on few clients. On huge number of requests server 1 would be faster as it interacts with 1 client only		
Process/Thread creation	The creation of processes takes more time than threads internally. Hence for huge number of clients , server3 would be preferred over server2 incase we need multiple service		
Memory usage	No extra memory used	Extra memory used	Memory is shared

Possible use cases based on benefits

- Server 1 can be used when there are less clients and waiting is allowed
- Server 2 uses multiple processes using fork and hence it takes more time (in process creation) and memory. However it helps in isolation of process , as in the code ,the parent handles socket connection while child handles interaction with client. So in case we need better process isolation and program which is easier to debug we can use server 2
- Server 3 uses multi threads which share memory and are faster too. Hence they can be used according to requirement