

LẬP TRÌNH MẠNG

BÁO CÁO HOMEWORK 03

Sinh viên thực hiện:
1712425 | Nguyễn Minh Hiền



*Môn : Lập trình mạng
Khoa : Công nghệ thông tin
Đại học Khoa học tự nhiên TP HCM*

Contents

1 Evaluate the achieved requirements.....	3
2 Idea of implementation.....	3
3 Specific program.....	4
3.1 Client.....	4
3.2 Server1.....	4
3.3 Netdclient.....	5
3.4 Netdsrv.....	6
3.5 Server2.....	7
4 Testing.....	8
5 Resource.....	12

1 Evaluate the requirements achieved

Categories	Completion level
Client	100%
Server1	100%
Netdclient	100%
Netdsrv	100%
Server2	100%

2 Idea of implementation

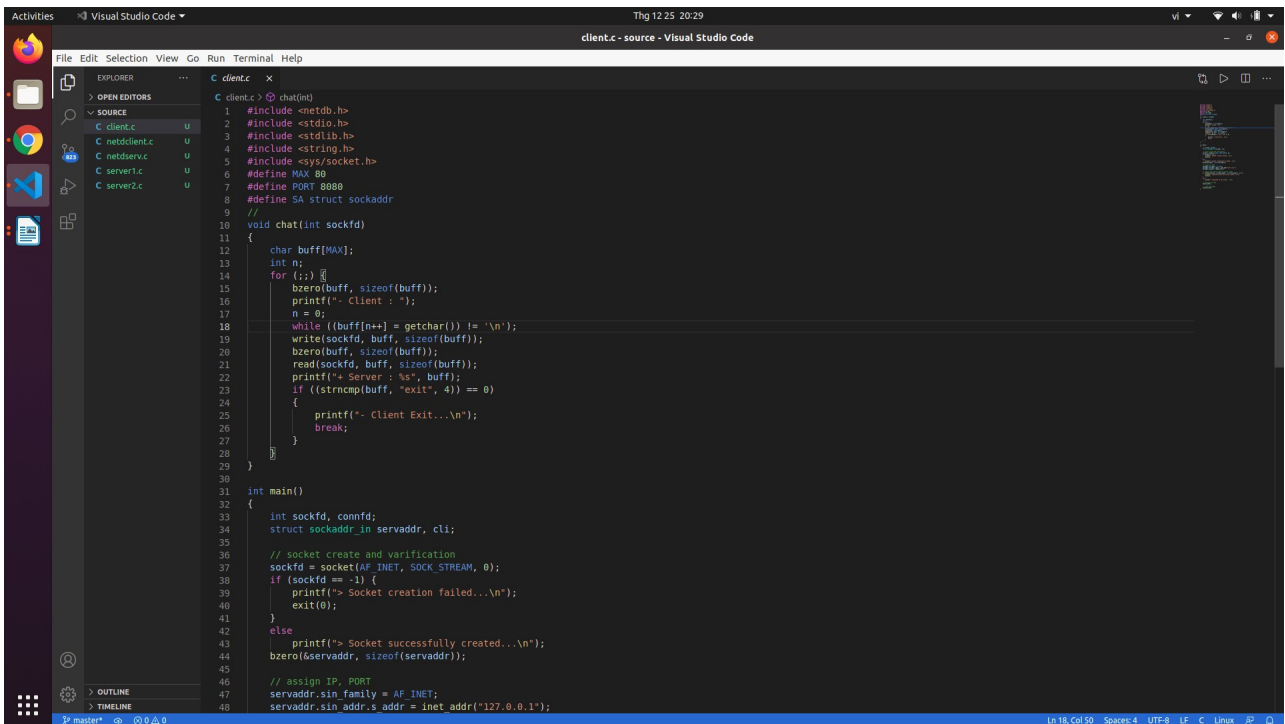
- Exercises based on knowledge of message queues and sockets
- First, we have to build a socket between **client** and **server1** to receive messages sent from **client**
- Next is to put this message in the queue
- Next, build the **netdclient** program to read messages in the queue
- Next, build a socket between **netdclient** and **netdsrv** to transmit messages. **Netdclient** is a client, **netdsrv** is a server
- Building **netdsrv** is both a server to receive messages and to put messages into a queue

- Finally, **server2** has the function of reading messages in the queue and displaying it on the screen

3 Specific program

3.1 Client

- Port: 8080



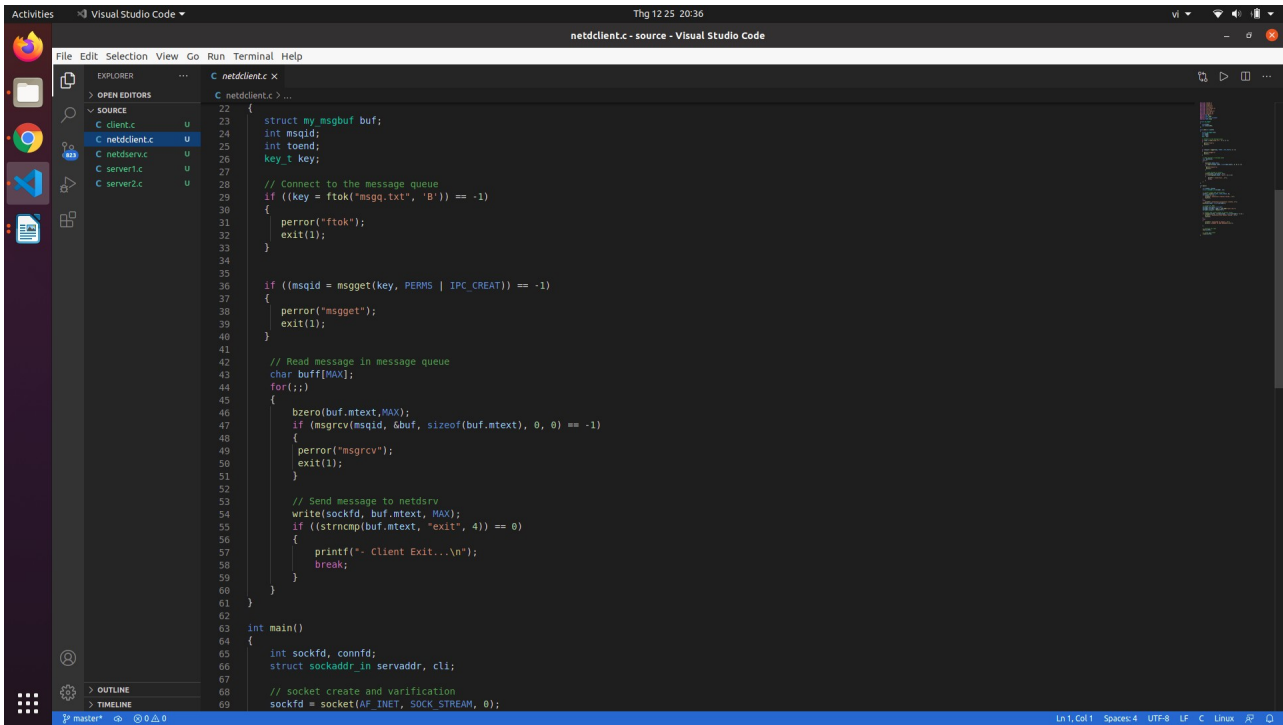
```
1 #include <netdb.h>
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include <string.h>
5 #include <sys/socket.h>
6 #define MAX 80
7 #define PORT 8080
8 #define SA struct sockaddr
9 //
10 void chat(int sockfd)
11 {
12     char buff[MAX];
13     int n;
14     for (;;) {
15         bzero(buff, sizeof(buff));
16         printf(" Client : ");
17         n = 0;
18         while ((buff[n++] = getchar()) != '\n')
19             write(sockfd, buff, sizeof(buff));
20         bzero(buff, sizeof(buff));
21         read(sockfd, buff, sizeof(buff));
22         printf(" Server : %s", buff);
23         if ((strcmp(buff, "exit", 4)) == 0)
24             {
25                 printf(" Client Exit...\n");
26                 break;
27             }
28     }
29 }
30
31 int main()
32 {
33     int sockfd, connfd;
34     struct sockaddr_in servaddr, cli;
35
36     // Socket create and verification
37     sockfd = socket(AF_INET, SOCK_STREAM, 0);
38     if (sockfd == -1) {
39         printf(" Socket creation failed...\n");
40         exit(0);
41     }
42     else
43         printf(" Socket successfully created...\n");
44     bzero(servaddr, sizeof(servaddr));
45
46     // assign IP, PORT
47     servaddr.sin_family = AF_INET;
48     servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
```

3.2 Server1

- Port: 8080
- We can think of PERM as the port to create message queue and connect



- Read the message in the message queue and send it to **netdsrv**
- * Port : 8000



```
22 {
23     struct my_msgbuf buf;
24     int msqid;
25     int toend;
26     key_t key;
27
28     // Connect to the message queue
29     if ((key = ftok("msgq.txt", 'B')) == -1)
30     {
31         perror("ftok");
32         exit(1);
33     }
34
35     if ((msqid = msgget(key, PERMS | IPC_CREAT)) == -1)
36     {
37         perror("msgget");
38         exit(1);
39     }
40
41     // Read message in message queue
42     char buff[MAX];
43     for(;;)
44     {
45         bzero(buf.mtext, MAX);
46         if (msgrcv(msqid, &buf, sizeof(buf.mtext), 0, 0) == -1)
47         {
48             perror("msgrcv");
49             exit(1);
50         }
51
52         // Send message to netdsrv
53         writetsockfd, buf.mtext, MAX);
54         if ((strcmp(buf.mtext, "exit", 4)) == 0)
55         {
56             printf("- Client Exit...\n");
57             break;
58         }
59     }
60 }
61
62
63 int main()
64 {
65     int sockfd, connfd;
66     struct sockaddr_in servaddr, cli;
67
68     // socket create and verification
69     sockfd = socket(AF_INET, SOCK_STREAM, 0);
```

3.4 Netdsrv

- Port: 8000
- Receive message from **netdclient** and write it to message queue2
- * PERMS: 0666

```

1  struct my_msgbuf buf;
2
3  // Create message queue 2
4  system("touch msgq2.txt");
5  if ((key = ftok("msgq2.txt", 'B')) == -1)
6  {
7      perror("ftok");
8      exit(1);
9  }
10
11  if ((msqid = msgget(key, PERMS2 | IPC_CREAT)) == -1)
12  {
13      perror("msgget");
14      exit(1);
15  }
16
17  buf.mtype = 1;
18
19  // Receive message from client & Write in message queue 2
20  for(;;)
21  {
22      bzero(buf, MAX);
23      bzero(buf.mtext, MAX);
24
25      // read the message from client and copy it in buffer
26      read(sockfd, buf, sizeof(buf));
27
28      // Write message in message queue 2
29      strcpy(buf.mtext, buf);
30      len = strlen(buf.mtext);
31      /* remove newline at end, if it exists */
32      if (buf.mtext[len-1] == '\n') buf.mtext[len-1] = '\0';
33      if (msgsnd(msqid, &buf, len+1, 0) == -1) /* +1 for '\0' */
34          perror("msgsnd");
35      if (strcmp("exit", buf, 4) == 0)
36      {
37          printf("Netdsvr exit...\n");
38          break;
39      }
40  }
41
42  int main()
43  {
44      int sockfd, connfd, len;
45      struct sockaddr_in servaddr, cli;

```

3.5 Server2

- Read the message in message queue 2 and display it on the screen

```

1  long mtype;
2  char mtext[200];
3
4  int main(void)
5  {
6      struct my_msgbuf buf;
7      int msqid;
8      int toend;
9      key_t key;
10
11      // Connect to Message queue 2
12      if ((key = ftok("msgq2.txt", 'B')) == -1)
13      {
14          perror("ftok");
15          exit(1);
16      }
17
18      if ((msqid = msgget(key, PERMS2 | IPC_CREAT)) == -1)
19      {
20          perror("msgget");
21          exit(1);
22      }
23
24      printf("> Message queue: ready to receive messages \n");
25
26      // Read message in message queue 2
27      for(;;)
28      {
29          bzero(buf.mtext, sizeof(buf.mtext));
30          if (msgrcv(msqid, &buf, sizeof(buf.mtext), 0, 0) == -1)
31          {
32              perror("msgrcv");
33              exit(1);
34          }
35
36          //
37          printf("> Message: %s\n", buf.mtext);
38          toend = strcmp(buf.mtext, "end");
39          if (toend == 0)
40              break;
41      }
42
43      printf("> Message queue: done receiving messages \n");
44      system("rm msgq2.txt");
45      return 0;
46  }

```

4 Testing

The order of running the programs is as follows

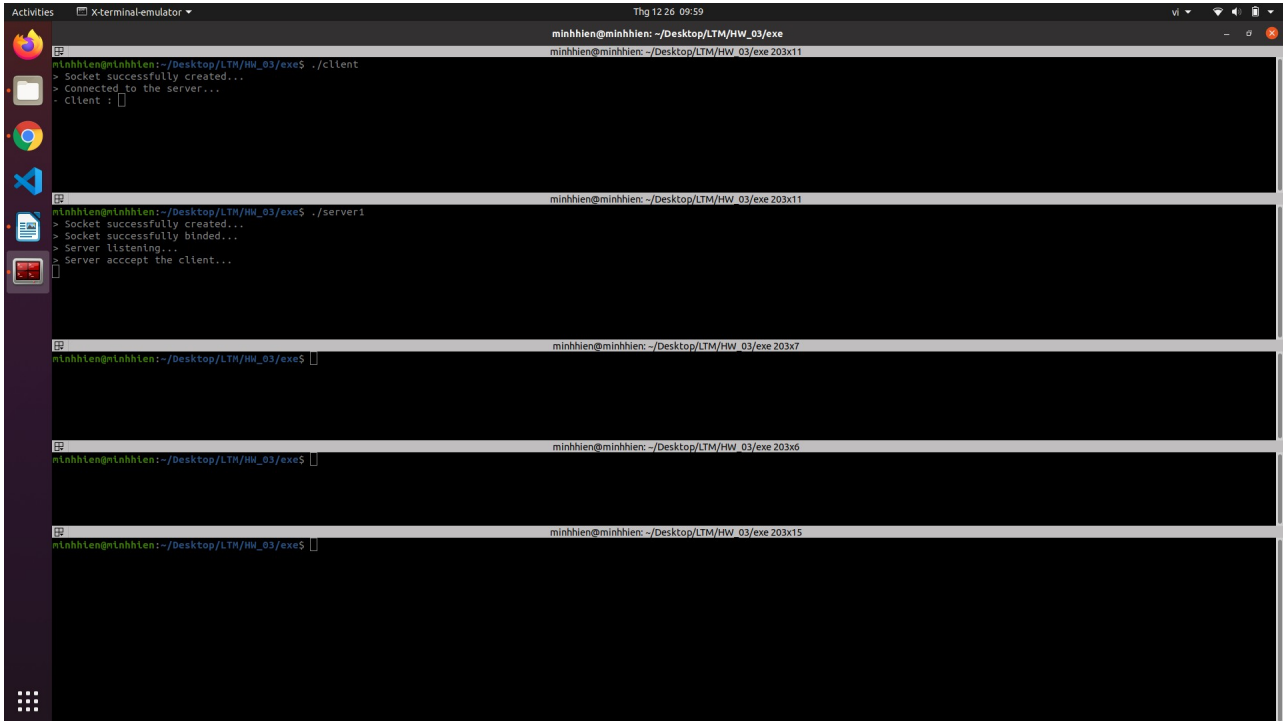
Server1 → Client → Netdsrv → Netdclient → Server2

```
minhhien@minhhien: ~/Desktop/LTM/HW_03/exe$ ./client
> Socket successfully created...
> Connected to the server...
- Client :

minhhien@minhhien: ~/Desktop/LTM/HW_03/exe$ ./server1
> Socket successfully created...
> Socket successfully binded...
> Server listening...
> Server accept the client...

minhhien@minhhien: ~/Desktop/LTM/HW_03/exe$

minhhien@minhhien: ~/Desktop/LTM/HW_03/exe$
```

The screenshot shows a terminal window titled "Xterminal-emulator" with a dark background. The terminal displays the following output:

```
minhhien@minhhien: ~/Desktop/LTM/HW_03/exe
minhhien@minhhien: ~/Desktop/LTM/HW_03/exe 203x11

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./client
> Socket successfully created...
> Connected to the server...
- Client : []

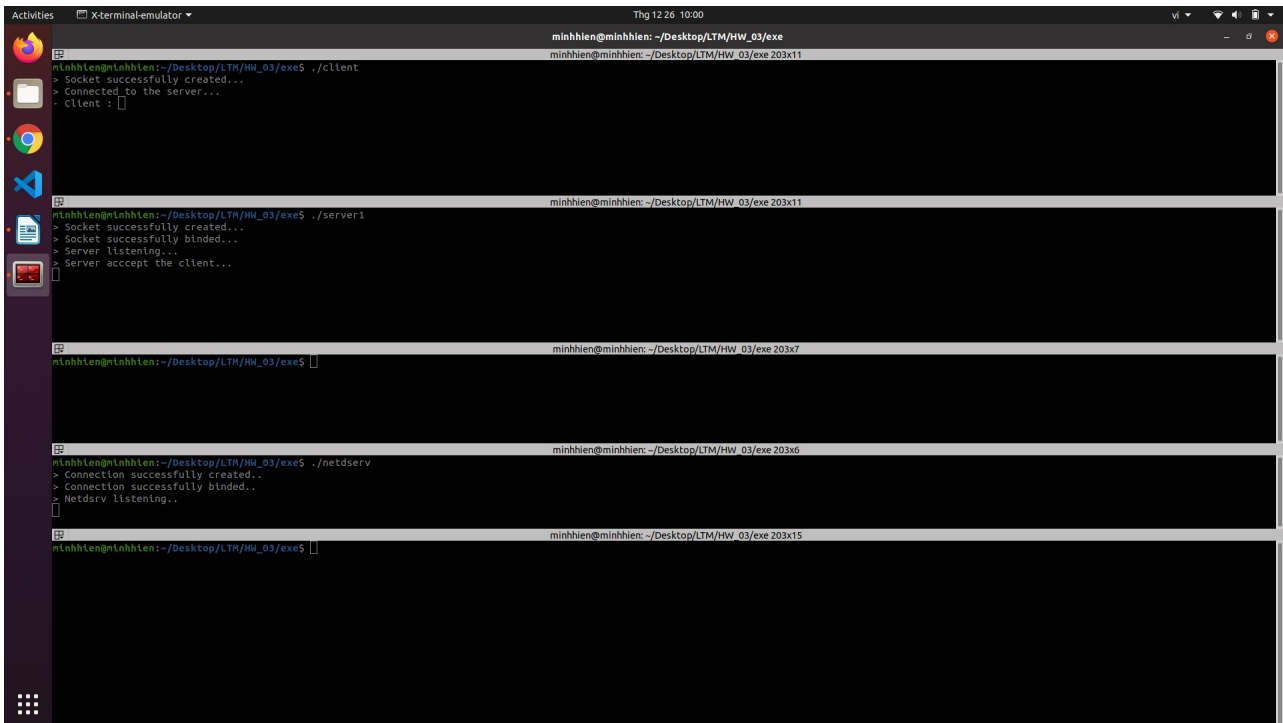
minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./server1
> Socket successfully created...
> Socket successfully binded...
> Server listening...
> Server accept the client...

minhhien@minhhien:~/Desktop/LTM/HW_03/exe 203x7

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$
```



The screenshot shows a terminal window titled "Xterminal-emulator" with a dark background. The terminal displays the following output:

```
minhhien@minhhien: ~/Desktop/LTM/HW_03/exe
minhhien@minhhien: ~/Desktop/LTM/HW_03/exe 203x11

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./client
> Socket successfully created...
> Connected to the server...
- Client : []

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./server1
> Socket successfully created...
> Socket successfully binded...
> Server listening...
> Server accept the client...

minhhien@minhhien:~/Desktop/LTM/HW_03/exe 203x7

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./netdsvr
> Connection successfully created..
> Connection successfully binded..
> Netdsvr listening..

minhhien@minhhien:~/Desktop/LTM/HW_03/exe 203x15

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$
```



The screenshot shows a terminal window titled "X-terminal-emulator" with a system clock of "Thg 12 26 10:01". The terminal displays a series of commands and their outputs, simulating a client-server interaction. The user is identified as "minhhien" and the current directory is "~/Desktop/LTM/HW_03/exe".

```
minhhien@minhhien: ~/Desktop/LTM/HW_03/exe
minhhien@minhhien: ~/Desktop/LTM/HW_03/exe
minhhien@minhhien: ~/Desktop/LTM/HW_03/exe$ ./client
> Socket successfully created...
> Connected to the server...
- client : []

minhhien@minhhien: ~/Desktop/LTM/HW_03/exe$ ./server1
> Socket successfully created...
> Socket successfully binded...
> Server listening...
> Server accepted the client...

minhhien@minhhien: ~/Desktop/LTM/HW_03/exe$ ./netdclient
> Connection successfully created..
> Connected to netdrv..
***READY TO SEND MESSAGE***

minhhien@minhhien: ~/Desktop/LTM/HW_03/exe$
> Connection successfully created..
> Connection successfully binded..
> Netdrv listening..
> Netdrv accepted Netdclient...
***READY TO RECEIVE MESSAGE***

minhhien@minhhien: ~/Desktop/LTM/HW_03/exe$
```



```
minhhien@minhhien: ~/Desktop/LTM/HW_03/exe
minhhien@minhhien: ~/Desktop/LTM/HW_03/exe 203x11
minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./client
> Socket successfully created...
> Connected to the server...
- Client : []

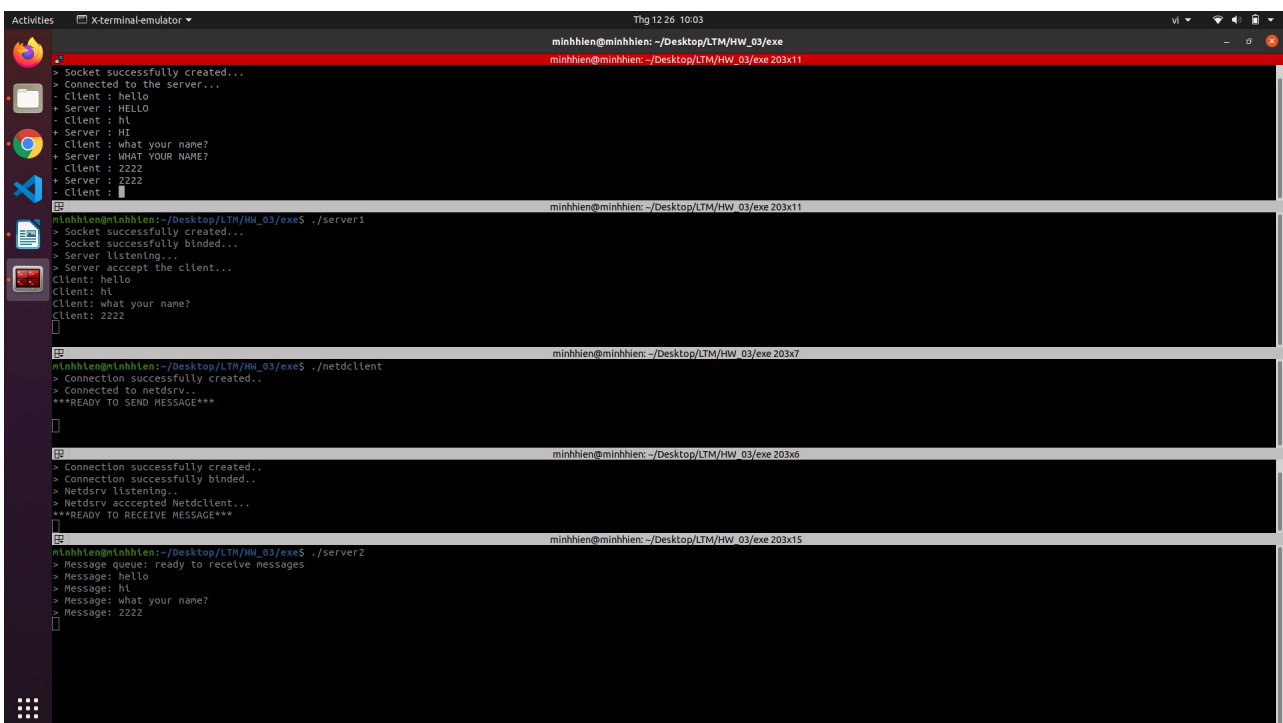
minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./server1
> Socket successfully created...
> Socket successfully binded...
> Server listening...
> Server accept the client...

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./netdclient
> Connection successfully created..
> Connected to netdsvr...
***READY TO SEND MESSAGE***

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./server2
> Connection successfully created..
> Connection successfully binded..
> Netdsvr listening..
> Netdsvr accepted Netdclient...
***READY TO RECEIVE MESSAGE***

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./server2
> Message queue: ready to receive messages
```

* Result:



```
minhhien@minhhien: ~/Desktop/LTM/HW_03/exe
minhhien@minhhien: ~/Desktop/LTM/HW_03/exe 203x11
minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./client
> Socket successfully created...
> Connected to the server...
- Client : hello
+ Server : HELLO
- Client : hi
+ Server : HI
- Client : what your name?
+ Server : WHAT YOUR NAME?
- Client : 2222
+ Server : 2222
- Client : []

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./server1
> Socket successfully created...
> Socket successfully binded...
> Server listening...
> Server accept the client...
Client: hello
Client: hi
Client: what your name?
Client: 2222

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./netdclient
> Connection successfully created..
> Connected to netdsvr..
***READY TO SEND MESSAGE***

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./server2
> Connection successfully created..
> Connection successfully binded..
> Netdsvr listening..
> Netdsvr accepted Netdclient...
***READY TO RECEIVE MESSAGE***

minhhien@minhhien:~/Desktop/LTM/HW_03/exe$ ./server2
> Message queue: ready to receive messages
> Message: hello
> Message: hi
> Message: what your name?
> Message: 2222
```

5 Resource

1. <https://www.cloudamqp.com/blog/2014-12-03-what-is-message-queuing.html>
2. https://www.tutorialspoint.com/inter_process_communication/inter_process_communication_message_queues.htm
3. <https://www.geeksforgeeks.org/tcp-server-client-implementation-in-c/>