SSN COLLEGE OF ENGINEERING, KALAVAKKAM
(An Autonomous Institution, Affiliated to Anna University, Chennai)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NETWORKS LAB
EXERCISE 2

Name: Jayannthan P T *Dept: CSE 'A'* *Roll No.: 205001049*

Aim:

Develop a socket program to establish a client server communication. The client sends data to server. The server in turn sends the message back to the client. Send multiple lines of text.

Algorithm:

SERVER

1. Create a socket using socket() system call.
2. Bind() is used to bind the socket with a specified address defined by sockaddr_in pointer, with the address, family, port set accordingly, bzero() is used to clear the address pointer initially.
3. listen() to make the created socket listen for incoming connections, maximum no of connections that can be accepted is specified here.
4. accept() call to make the server accept any connection requests, the parameter sockaddr_in accept() holds the requesting clients address, with which the messages are addressed to.
5. Read() is used to read data from client into a temporary buffer.
6. Sends the same message back to the client using write() system call.
7. Then the received message is displayed.

CLIENT

1. Create a socket using socket() system call.
2. The socket descriptor is noted using which a connect() call is made to connect to server, whose address is to be specified as a pointer of sockaddr_in in the sockaddr field of connect().
3. The sockaddr_in pointer holds the address, set by user, some port, ip of the server machine, respectively.

4. Once the server accepts the call, the client requests input from the user, sends it to server with the write() call. The server returns the same message and it is read using read() system call.

Code:

Server

```
#include <netinet/in.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/socket.h>
#include <string.h>
#include <sys/types.h>

int main(int argc, char const* argv[])
{
    int port=atoi(argv[1]);
    int serSockID = socket(AF_INET, SOCK_STREAM, 0);
    char serMsg[255] = "Hello Client";
    struct sockaddr_in servAddr;
    servAddr.sin_family = AF_INET;
    servAddr.sin_port = htons(port);
    servAddr.sin_addr.s_addr = INADDR_ANY;
    bind(serSockID, (struct sockaddr*)&servAddr,sizeof(servAddr));
    listen(serSockID, 1);
    int clientSocket = accept(serSockID, NULL, NULL);
    printf("\nConnected\n");
    int connected=1;
    while(connected)
    {
        read(clientSocket,serMsg,sizeof(serMsg));
        printf("Message recieved: %s\n!!!Sending message again to client!!!\n\n",serMsg);
        send(clientSocket, serMsg, sizeof(serMsg), 0);
        if(strcmp(serMsg,"exit")==0)
        {
            connected =0;
        }
    }
    return 0;
}
```

Client

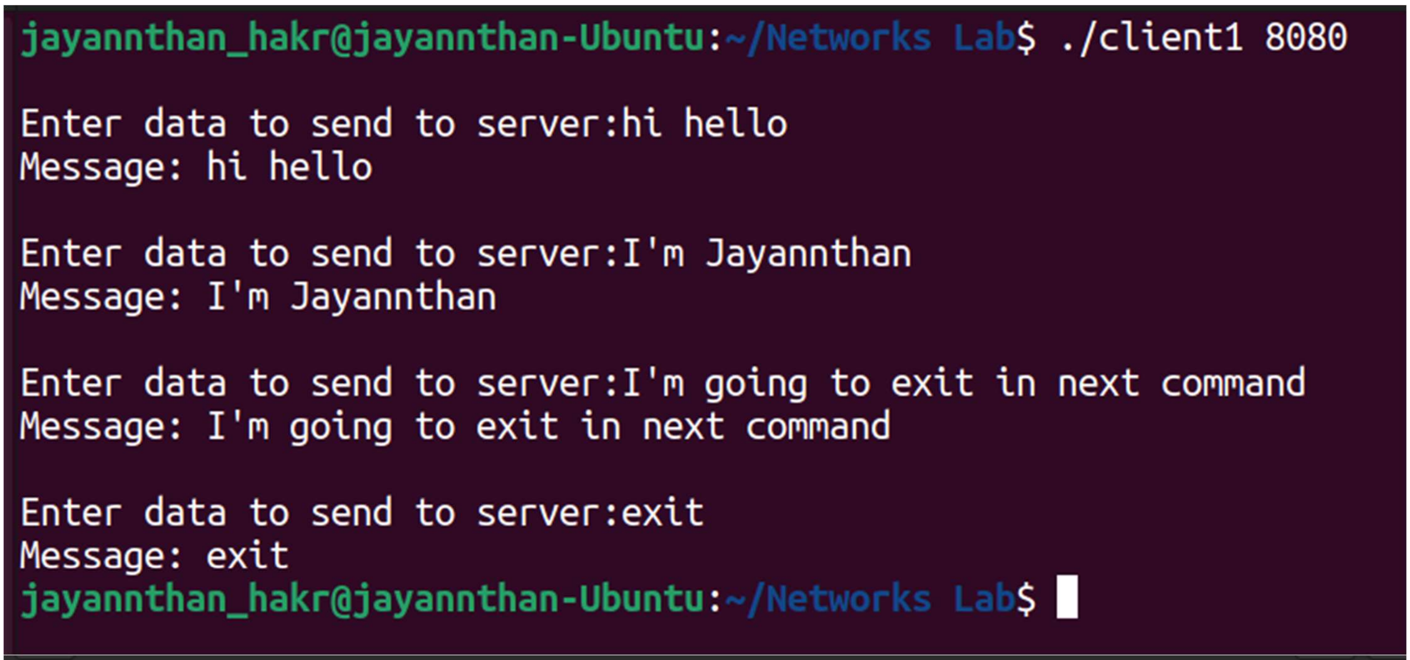
```
#include <netinet/in.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/socket.h>
#include <sys/types.h>

int main(int argc, char const* argv[])
{
```

```c
    int port=atoi(argv[1]);
    int sockD = socket(AF_INET, SOCK_STREAM, 0);
    struct sockaddr_in servAddr;
    servAddr.sin_family = AF_INET;
    servAddr.sin_port= htons(port);
    servAddr.sin_addr.s_addr = INADDR_ANY;
    int connectStatus= connect(sockD, (struct sockaddr*)&servAddr,sizeof(servAddr));
    //printf("\nConnected\n");
    if (connectStatus == -1) {
        printf("Error...\n");
    }
    else
    {
        int connected=1;
        char strData[255];
        while(connected)
        {

            printf("\nEnter data to send to server:");
            scanf(" %[^\n]s",strData);
            write(sockD,strData,sizeof(strData));
            recv(sockD, strData, sizeof(strData), 0);
            printf("Message: %s\n", strData);
            if(strcmp(strData,"exit")==0)
            {
                connected =0;
            }
        }
    }
    return 0;
}
```

Output:



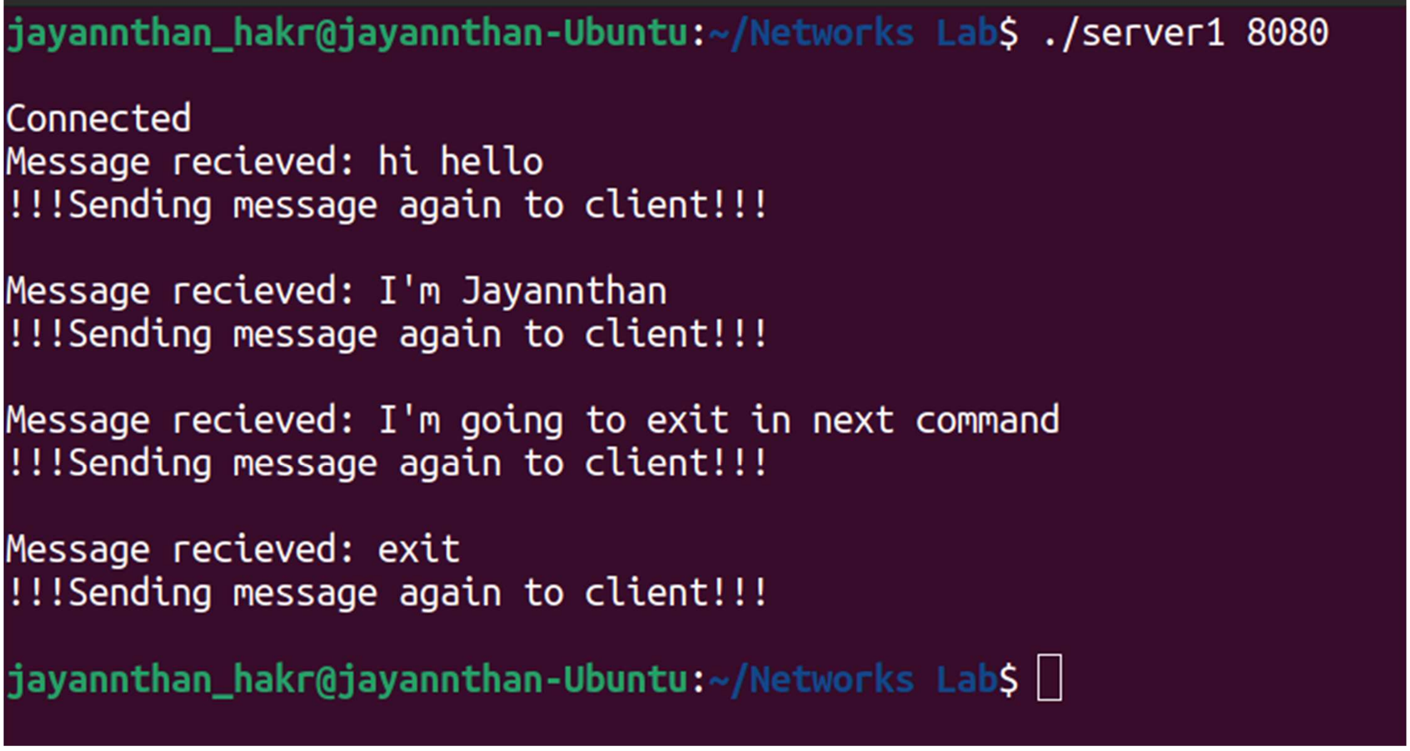
```
jayannthan_hakr@jayannthan-Ubuntu:~/Networks Lab$ ./client1 8080

Enter data to send to server:hi hello
Message: hi hello

Enter data to send to server:I'm Jayannthan
Message: I'm Jayannthan

Enter data to send to server:I'm going to exit in next command
Message: I'm going to exit in next command

Enter data to send to server:exit
Message: exit
jayannthan_hakr@jayannthan-Ubuntu:~/Networks Lab$ 
```



```
jayannthan_hakr@jayannthan-Ubuntu:~/Networks Lab$ ./server1 8080

Connected
Message recieved: hi hello
!!!Sending message again to client!!!

Message recieved: I'm Jayannthan
!!!Sending message again to client!!!

Message recieved: I'm going to exit in next command
!!!Sending message again to client!!!

Message recieved: exit
!!!Sending message again to client!!!

jayannthan_hakr@jayannthan-Ubuntu:~/Networks Lab$ 
```

Learning outcome:
Learnt to create connection using sockets
Learnt to communicate between server and client using socket