Computer networks

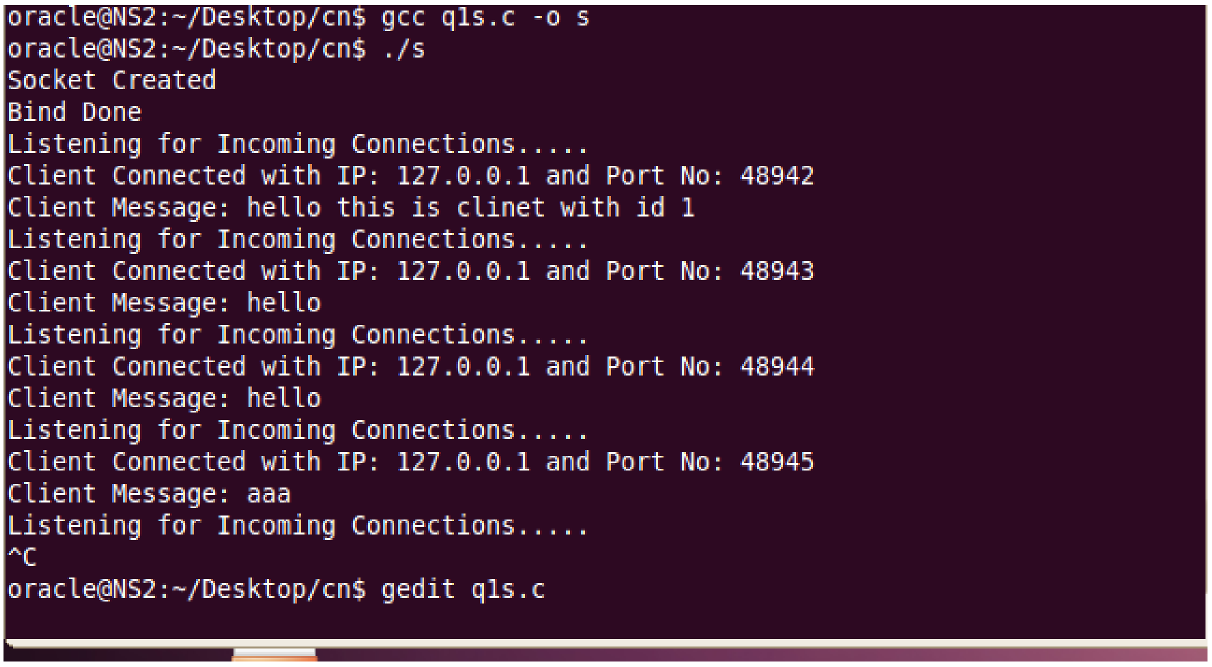
Lab 06

In Lab-Statement 01:

server.c:

```
#include <stdio.h>

#include <string.h>

#include <sys/socket.h> //socket

#include <arpa/inet.h> //inet_addr


int main(void)

{

 int socket_desc, client_sock, client_size;

        struct sockaddr_in server_addr, client_addr;

        int client_s;

        char server_message[2000], client_message[2000];                // Sending values from the server and receive
from the server we need this

        char hello[]="hello i am server. Your received id is  "; // id will be on 41th index and size of this array is 40

        //Cleaning the Buffers


        memset(server_message,'\0',sizeof(server_message));

        memset(client_message,'\0',sizeof(client_message));     // Set all bits of the padding field//


        //Creating Socket


        socket_desc = socket(AF_INET, SOCK_STREAM, 0);


        if(socket_desc < 0)

        {

                printf("Could Not Create Socket. Error!!!!!\n");

                return -1;

        }
	printf("Socket Created\n");


	//Binding IP and Port to socket


	server_addr.sin_family = AF_INET;		/* Address family = Internet */

	server_addr.sin_port = htons(2000);		// Set port number, using htons function to use proper byte
order */

	server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");    /* Set IP address to localhost */

	if(bind(socket_desc, (struct sockaddr*)&server_addr, sizeof(server_addr))<0)    // Bind the address struct
to the socket.  /

					//bind() passes file descriptor, the address structure,and the length of the address
structure

	{

		printf("Bind Failed. Error!!!!!\n");

		return -1;

	}


	printf("Bind Done\n");


	//Put the socket into Listening State

	do{

	if(listen(socket_desc, 1) < 0)				//This listen() call tells the socket to listen to the incoming
connections.

    // The listen() function places all incoming connection into a "backlog queue" until accept() call accepts the
connection.

	{

		printf("Listening Failed. Error!!!!!\n");

		return -1;

	}


	printf("Listening for Incoming Connections.....\n");


	//Accept the incoming Connections
```

```
        client_size = sizeof(client_addr);




        client_sock = accept(socket_desc, (struct sockaddr*)&client_addr, &client_size);         // heree particular
client k liye new socket create kr rhaa ha


        if (client_sock < 0)
        {
            printf("Accept Failed. Error!!!!!!\n");
            return -1;
        }


        printf("Client Connected with IP: %s and Port No:
%i\n",inet_ntoa(client_addr.sin_addr),ntohs(client_addr.sin_port));
                                //inet_ntoa() function converts the Internet host address in, given in network byte order,
to a string in IPv4 dotted-decimal notation


        //Receive the message from the client


        if (recv(client_sock, client_message, sizeof(client_message),0) < 0)
        {
            printf("Receive Failed. Error!!!!!\n");
            return -1;
        }


        printf("Client Message: %s\n",client_message);


        //Send the message back to client


        strcpy(server_message,hello);
        client_s=strlen(client_message)-1;
        server_message[39]=client_message[client_s];
        if (send(client_sock, server_message, strlen(server_message),0)<0)

        {

                printf("Send Failed. Error!!!!!\n");

                return -1;

        }


        memset(server_message,'\0',sizeof(server_message));

        memset(client_message,'\0',sizeof(client_message));


        }while(1);

        //Closing the Socket


        close(client_sock);

        close(socket_desc);

return 0;

}
```



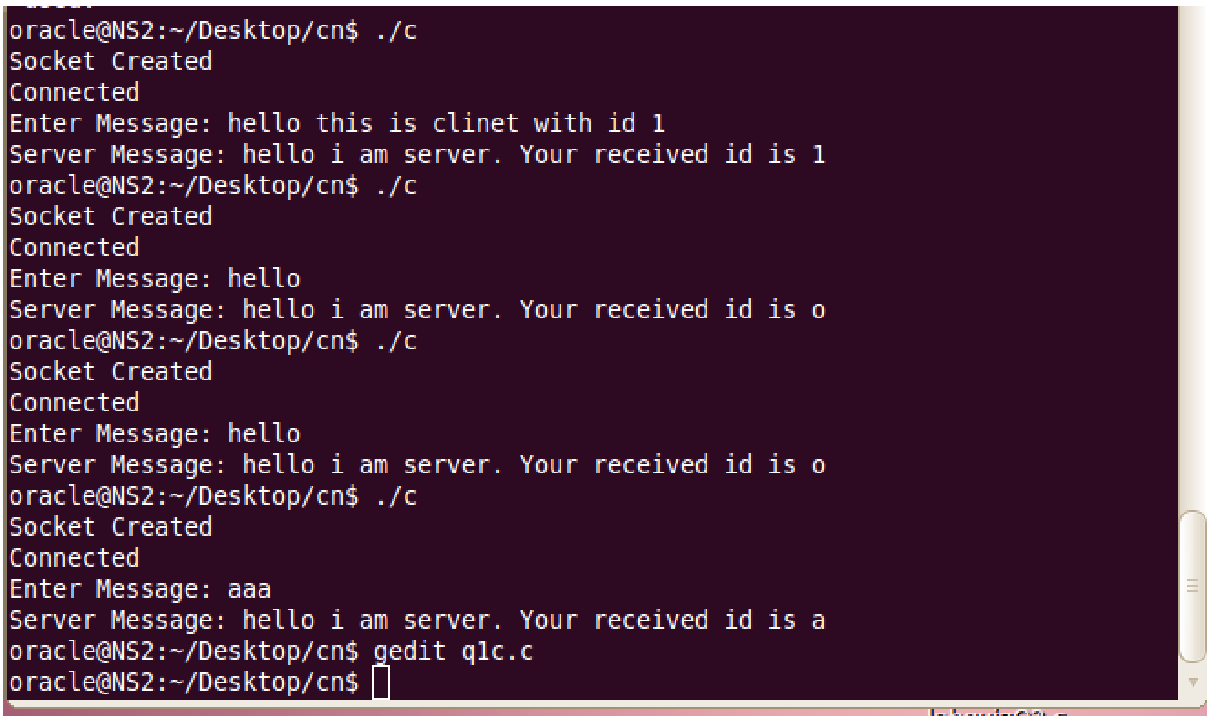
```
oracle@NS2:~/Desktop/cn$ gcc q1s.c -o s
oracle@NS2:~/Desktop/cn$ ./s
Socket Created
Bind Done
Listening for Incoming Connections.....
Client Connected with IP: 127.0.0.1 and Port No: 48942
Client Message: hello this is clinet with id 1
Listening for Incoming Connections.....
Client Connected with IP: 127.0.0.1 and Port No: 48943
Client Message: hello
Listening for Incoming Connections.....
Client Connected with IP: 127.0.0.1 and Port No: 48944
Client Message: hello
Listening for Incoming Connections.....
Client Connected with IP: 127.0.0.1 and Port No: 48945
Client Message: aaa
Listening for Incoming Connections.....
^C
oracle@NS2:~/Desktop/cn$ gedit q1s.c
```

client.c:

```
#include <stdio.h>
```

```
#include <string.h>

#include <sys/socket.h> //socket

#include <arpa/inet.h> //inet_addr


int main(void)

{

    int socket_desc;

    struct sockaddr_in server_addr;

    char server_message[2000], client_message[2000];


    //Cleaning the Buffers


    memset(server_message,'\0',sizeof(server_message));

    memset(client_message,'\0',sizeof(client_message));


    //Creating Socket


    socket_desc = socket(AF_INET, SOCK_STREAM, 0);


    if(socket_desc < 0)

    {

        printf("Could Not Create Socket. Error!!!!!\n");

        return -1;

    }


    printf("Socket Created\n");


    //Specifying the IP and Port of the server to connect


    server_addr.sin_family = AF_INET;

    server_addr.sin_port = htons(2000);
    server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");


    //Now connecting to the server accept() using connect() from client side


    if(connect(socket_desc, (struct sockaddr*)&server_addr, sizeof(server_addr)) < 0)
    {
        printf("Connection Failed. Error!!!!!");
        return -1;
    }


    printf("Connected\n");


    //Get Input from the User


    printf("Enter Message: ");
    gets(client_message);                              //One is that gets() will only get character string data.
                                       //                         will get only one variable at a time.

                                  //  reads characters from stdin and loads them into str
    //Send the message to Server
    if(send(socket_desc, client_message, strlen(client_message),0) < 0)
    {
        printf("Send Failed. Error!!!!\n");
        return -1;
    }


    //Receive the message back from the server


    if(recv(socket_desc, server_message, sizeof(server_message),0) < 0)
    {
        printf("Receive Failed. Error!!!!!\n");
```

```
        return -1;
    }

    printf("Server Message: %s\n",server_message);

    memset(server_message,'\0',sizeof(server_message));
    memset(client_message,'\0',sizeof(client_message));

    //Closing the Socket

    close(socket_desc);

    return 0;
}
```



```
oracle@NS2:~/Desktop/cn$ ./c
Socket Created
Connected
Enter Message: hello this is clinet with id 1
Server Message: hello i am server. Your received id is 1
oracle@NS2:~/Desktop/cn$ ./c
Socket Created
Connected
Enter Message: hello
Server Message: hello i am server. Your received id is o
oracle@NS2:~/Desktop/cn$ ./c
Socket Created
Connected
Enter Message: hello
Server Message: hello i am server. Your received id is o
oracle@NS2:~/Desktop/cn$ ./c
Socket Created
Connected
Enter Message: aaa
Server Message: hello i am server. Your received id is a
oracle@NS2:~/Desktop/cn$ gedit q1c.c
oracle@NS2:~/Desktop/cn$
```

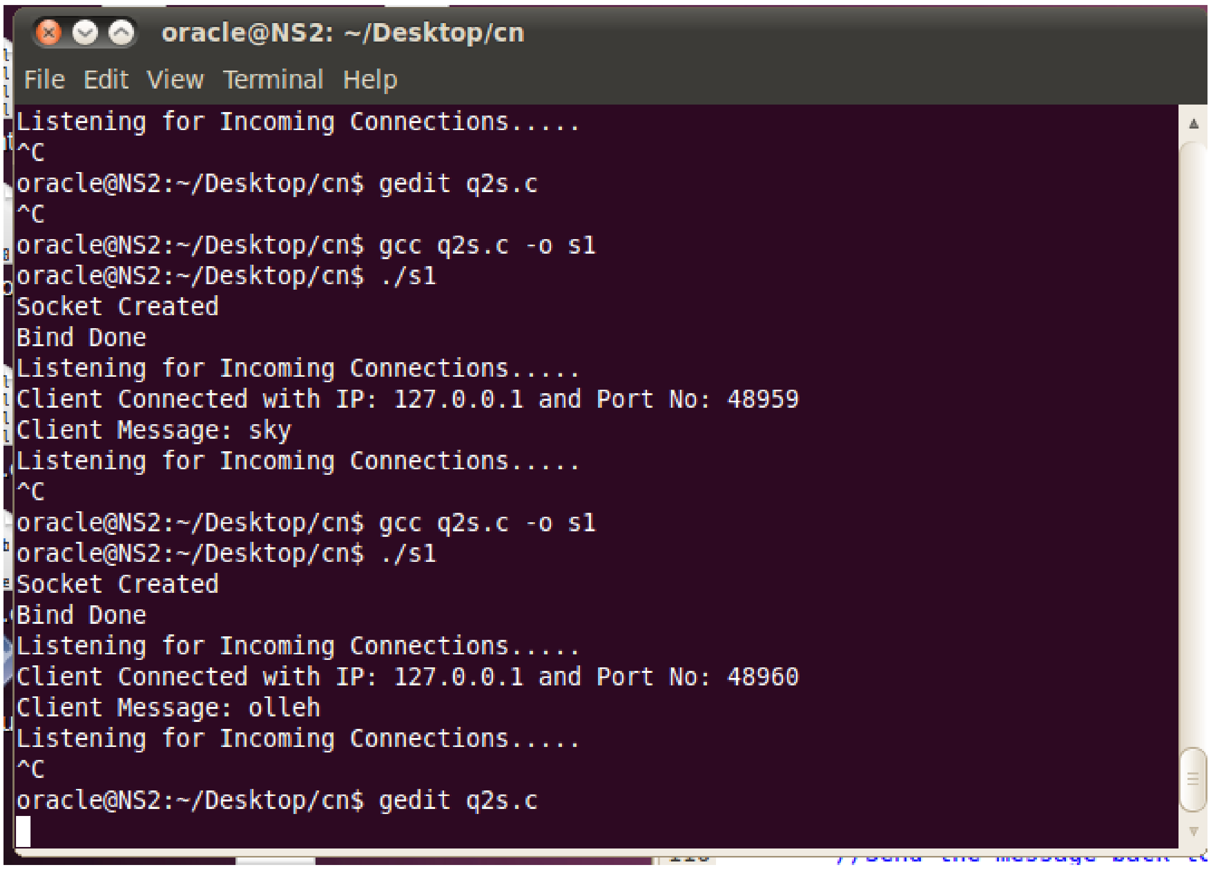
In Lab-Statement 02:

server.c:

```
#include <stdio.h>

#include <string.h>

#include <sys/socket.h> //socket

#include <arpa/inet.h> //inet_addr


int main(void)

{

	int socket_desc, client_sock, client_size;

	struct sockaddr_in server_addr, client_addr;		//SERVER ADDR will have all the server address

	char server_message[2000], client_message[2000];		// Sending values from the server
and receive from the server we need this


	//Cleaning the Buffers


	memset(server_message,'\0',sizeof(server_message));

	memset(client_message,'\0',sizeof(client_message));	// Set all bits of the padding field//


	//Creating Socket


	socket_desc = socket(AF_INET, SOCK_STREAM, 0);


	if(socket_desc < 0)

	{

		printf("Could Not Create Socket. Error!!!!!\n");

		return -1;

	}


	printf("Socket Created\n");
```

```
    //Binding IP and Port to socket


    server_addr.sin_family = AF_INET;              /* Address family = Internet */

    server_addr.sin_port = htons(2000);              // Set port number, using htons function to use
proper byte order */

    server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");    /* Set IP address to localhost */





                    // BINDING FUNCTION


    if(bind(socket_desc, (struct sockaddr*)&server_addr, sizeof(server_addr))<0)    // Bind the
address struct to the socket.  /

                                  //bind() passes file descriptor, the address structure,and the length of the
address structure

    {

        printf("Bind Failed. Error!!!!!\n");

        return -1;

    }


    printf("Bind Done\n");


    //Put the socket into Listening State

    do{

    if(listen(socket_desc, 1) < 0)                           //This listen() call tells the socket to listen to the
incoming connections.

   // The listen() function places all incoming connection into a "backlog queue" until accept() call
accepts the connection.

    {

        printf("Listening Failed. Error!!!!!\n");

        return -1;

    }
    printf("Listening for Incoming Connections.....\n");

    //Accept the incoming Connections

    client_size = sizeof(client_addr);



    client_sock = accept(socket_desc, (struct sockaddr*)&client_addr, &client_size);        // heree
particular client k liye new socket create kr rhaa ha

    if (client_sock < 0)
    {
        printf("Accept Failed. Error!!!!!!\n");
        return -1;
    }

    printf("Client Connected with IP: %s and Port No:
%i\n",inet_ntoa(client_addr.sin_addr),ntohs(client_addr.sin_port));
                    //inet_ntoa() function converts the Internet host address in, given in network
byte order, to a string in IPv4 dotted-decimal notation

    //Receive the message from the client

    if (recv(client_sock, client_message, sizeof(client_message),0) < 0)
    {
        printf("Receive Failed. Error!!!!!\n");
        return -1;
    }



    int i=0;
```

```
        int flag;

        int a,b;

        const char s[2]=" ";

        char*token;

        token=strtok(client_message,s);

        int l=sizeof(token);

        while(token!=NULL)

        {

        while(i<=sizeof(token))

        {

if(token[i]=='a'||token[i]=='e'||token[i]=='i'||token[i]=='o'||token[i]=='u'||token[i]=='A'||token[i]=
='E'||token[i]=='I'||token[i]=='O'||token[i]=='U')

        {flag=1;break;}

        else{flag=0;}

        i++;

        }

        i=0;

        if(flag==1)

        {

        for(a=0,b=l;a<=b;a++,b--)

        {

        char temp=token[a];

        token[a]=token[b];

        token[b]=temp;

        }


        }

        token=strtok(NULL,s);

        }


        printf("Client Message: %s\n",client_message);
        //Send the message back to client


        strcpy(server_message, client_message);


        if (send(client_sock, server_message, strlen(client_message),0)<0)
        {
                printf("Send Failed. Error!!!!!\n");
                return -1;
        }


        memset(server_message,'\0',sizeof(server_message));
        memset(client_message,'\0',sizeof(client_message));
              }while(1);
        //Closing the Socket


        close(client_sock);
        close(socket_desc);
        return 0;
}
```



```
Listening for Incoming Connections.....
^C
oracle@NS2:~/Desktop/cn$ gedit q2s.c
^C
oracle@NS2:~/Desktop/cn$ gcc q2s.c -o s1
oracle@NS2:~/Desktop/cn$ ./s1
Socket Created
Bind Done
Listening for Incoming Connections.....
Client Connected with IP: 127.0.0.1 and Port No: 48959
Client Message: sky
Listening for Incoming Connections.....
^C
oracle@NS2:~/Desktop/cn$ gcc q2s.c -o s1
oracle@NS2:~/Desktop/cn$ ./s1
Socket Created
Bind Done
Listening for Incoming Connections.....
Client Connected with IP: 127.0.0.1 and Port No: 48960
Client Message: olleh
Listening for Incoming Connections.....
^C
oracle@NS2:~/Desktop/cn$ gedit q2s.c
```

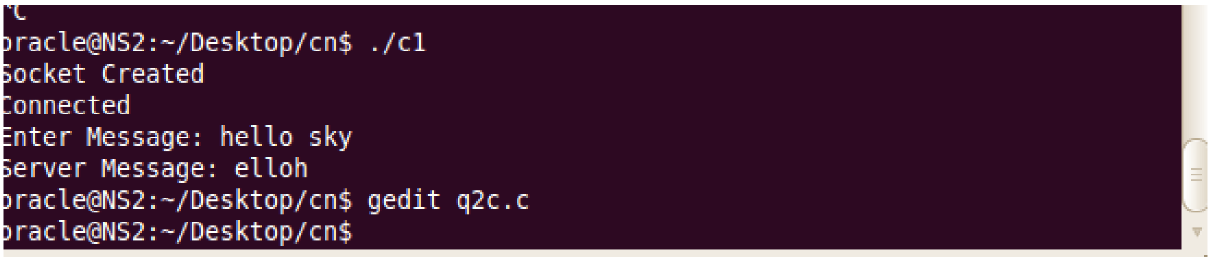
Client.c:

```
#include <stdio.h>

#include <string.h>

#include <sys/socket.h> //socket

#include <arpa/inet.h> //inet_addr


int main(void)

{

        int socket_desc;

        struct sockaddr_in server_addr;

        char server_message[2000], client_message[2000];


        //Cleaning the Buffers
```

```
	memset(server_message,'\0',sizeof(server_message));
	memset(client_message,'\0',sizeof(client_message));

	//Creating Socket

	socket_desc = socket(AF_INET, SOCK_STREAM, 0);

	if(socket_desc < 0)
	{
		printf("Could Not Create Socket. Error!!!!!\n");
		return -1;
	}

	printf("Socket Created\n");

	//Specifying the IP and Port of the server to connect

	server_addr.sin_family = AF_INET;
	server_addr.sin_port = htons(2000);
	server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");

	//Now connecting to the server accept() using connect() from client
side

	if(connect(socket_desc, (struct sockaddr*)&server_addr,
sizeof(server_addr)) < 0)
```

```
    {
        printf("Connection Failed. Error!!!!!");
        return -1;
    }

    printf("Connected\n");

    //Get Input from the User

    printf("Enter Message: ");
    gets(client_message);                    //One is that gets() will
only get character string data.
                                  //      will get only one variable at a
time.

                    //  reads characters from stdin and loads them into str
    //Send the message to Server

    if(send(socket_desc, client_message, strlen(client_message),0) < 0)
    {
        printf("Send Failed. Error!!!!\n");
        return -1;
    }

    //Receive the message back from the server

    if(recv(socket_desc, server_message, sizeof(server_message),0) < 0)
```

```
	{
		printf("Receive Failed. Error!!!!!\n");
		return -1;
	}
	int i=0;
	int flag;
	int a,b;
	const char s[2]=" ";
	char*token;
	token=strtok(server_message,s);
	int l=sizeof(token);
	while(token!=NULL)
	{
	while(i<=sizeof(token))
	{
if(token[i]!='a'||token[i]!='e'||token[i]!='i'||token[i]!='o'||token[i]!='u'||token[i
]!='A'||token[i]!='E'||token[i]!='I'||token[i]!='O'||token[i]!='U')
	{flag=1;break;}
	else{flag=0;}
	i++;
	}
	i=0;
	if(flag==1)
	{
	for(a=0,b=l-1;a<b;a++,b--)
	{
        char temp=token[a];
        token[a]=token[b];
        token[b]=temp;
        }


        }
        token=strtok(NULL,s);
        }
        printf("Server Message: %s\n",server_message);


        memset(server_message,'\0',sizeof(server_message));
        memset(client_message,'\0',sizeof(client_message));


        //Closing the Socket


        close(socket_desc);


        return 0;
}
```



```
^C
oracle@NS2:~/Desktop/cn$ ./c1
Socket Created
Connected
Enter Message: hello sky
Server Message: elloh
oracle@NS2:~/Desktop/cn$ gedit q2c.c
oracle@NS2:~/Desktop/cn$
```