Name: Brijesh Rohit

Admission no.: U19CS009

DISTRIBUTED SYSTEMS - ASSIGNMENT - 4

Implement echo client-server message passing application. Message sent from client should be displayed on server and then program should terminate.

1. Write a server (TCP) C Program that opens a listening socket and waits to serve client.

CODE=>

```
//U19CS009
//Brijesh Rohit

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

int main() {
//creating a socket for server
    int server_socket, client_socket;
    server_socket = socket(AF_INET, SOCK_STREAM, 0);
    if (server_socket < 0)
    {
        perror("--> Server error !!!\n");
        exit(1);
    }
//define server_address and client_address
    struct sockaddr_in server_address, client_address;
    server_address.sin_family = AF_INET;
    server_address.sin_port = htons(9002);
    //passing port number 9002
    server_address.sin_addr.s_addr = INADDR_ANY;
    //specifing local machine address
    //binding socket with specific IP and port number
    int bind_connection = bind(server_socket, (struct sockaddr
*)&server_address, sizeof(server_address));
    printf("Bind to the port number : 9002");
    //listening to connection
    listen(server_socket, 4);
    printf("\nListening...........\n");
    //accept a connection
    int size_client_addr = sizeof(client_address);
    client_socket = accept(server_socket, (struct sockaddr *)&client_address,
&size_client_addr);
    printf(".....Client is connected.....\n");
    //recive data from client
    char response_client[256];
    recv(client_socket, &response_client, sizeof(response_client), 0);
    //last parameter is optional so putting 0
```

```
    //print out the data received from client
    printf("Message from client : ");
    printf("%s", response_client);
    printf("\n");
    //close the socket
    close(server_socket);
    printf("\nClient disconnected……..\n");
    return 0;
}
```

2. Write a client (TCP) C Program that connects with the server program knowing IP address and port number.

CODE=>

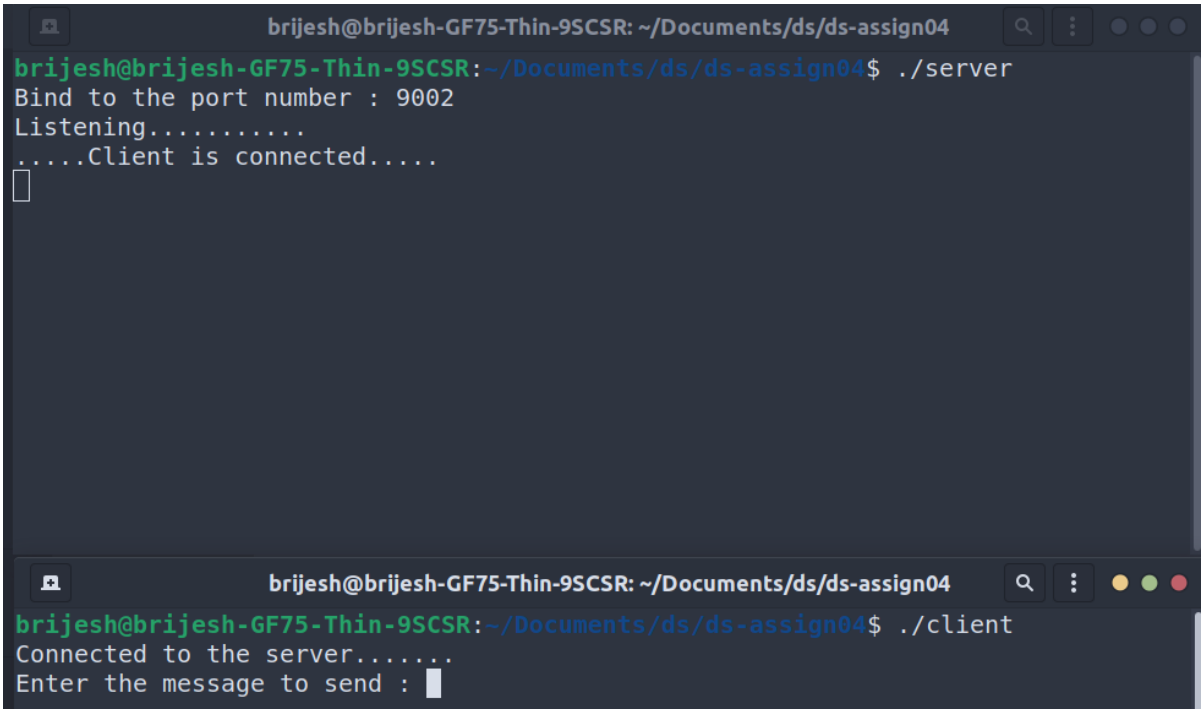
```
//U19CS009
//Brijesh Rohit

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

int main() {
//Creating a socket for client
    int no_socket;
    no_socket = socket(AF_INET, SOCK_STREAM, 0);
//specifing address for client socket
    struct sockaddr_in client_address;
    client_address.sin_family = AF_INET;
    client_address.sin_port = htons(9002);
//passing port number 9002
    client_address.sin_addr.s_addr = INADDR_ANY;
    //specifing local machine address
    int connection_status = connect(no_socket, (struct sockaddr
*)&client_address, sizeof(client_address));
    // 0 OK -1 error
//Checking whether there is an error in connection
    if (connection_status < 0)
    {
        perror("--->There was an error making connection with the remote
socket\n\n");
        exit(1);
    }
    printf("Connected to the server.......\n");
//send data to server
    char send_server[256];
    printf("Enter the message to send : ");
    scanf("%s", send_server);
    send(no_socket, send_server, sizeof(send_server), 0);
    printf("\n");
```

```
//close the connection
    close(no_socket);
    printf("\n Disconnected from server.......\n");
    return 0;
}
```

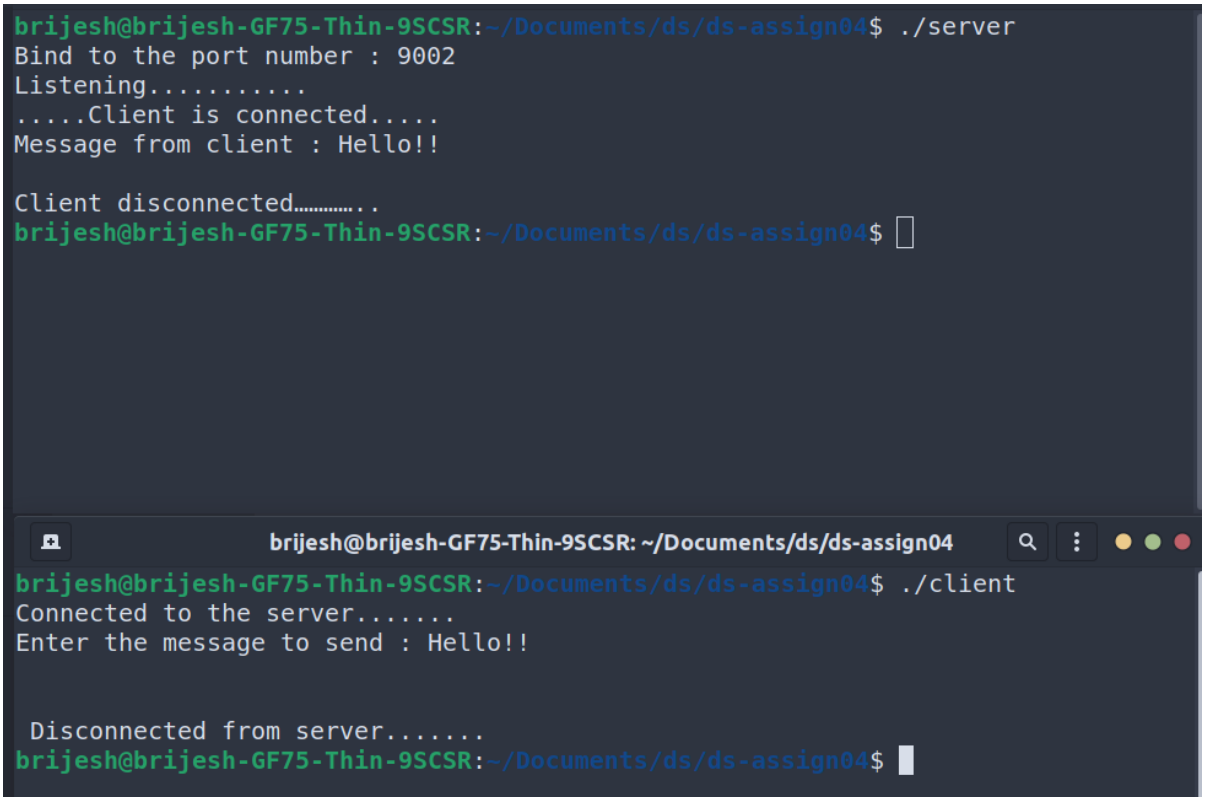
3. Get the input string from console on client and send it to server, server displays the same string.

Waiting for client to respond

Client connected and connection established



Requested completed and message exchanged, and network ended.



In case of error on client side, connection automatically ends and nothing is passed.