Assignment 3

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.
2. Write a client (TCP) C Program that connects with the server program knowing IP address and port number.
3. Get the input string from console on client and send it to server, server displays the same string.

Code for client:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>

int main()
{
    char *ip = "127.0.0.1";
    int port = 5000;

    int sock;
    struct sockaddr_in addr;
    socklen_t addr_size;
    char buffer[1024];
    int n;

    sock = socket(AF_INET, SOCK_STREAM, 0);
    if (n < 0)
    {
        perror("Socket error......");

printf("#######################################################\n");
        printf("\n");

        exit(1);
    }
    printf("TCP server socket created......\n");

    memset(&addr, '\0', sizeof(addr));
    addr.sin_family = AF_INET;
    addr.sin_port = port;
    addr.sin_addr.s_addr = inet_addr(ip);
```

```
    connect(sock, (struct sockaddr *)&addr, sizeof(addr));
    printf("Server connected......\n");

printf("#######################################################\n");
    printf("\n");

    while (1)
    {

        bzero(buffer, 1024);
        printf("Enter message to send a server (exit for quit) : \n");
        scanf("%[^\n]%*c", buffer);

        printf("Your message sended to server is : \n");
        printf("%s", buffer);
        printf("\n\n");

        send(sock, buffer, strlen(buffer), 0);

        if ((strncmp(buffer, "exit", 4)) == 0)
        {
            close(sock);
            printf("Server disconnected......\n");

printf("#######################################################\n");
            printf("\n");
            exit(0);
        }
    }

    return 0;
}
```

Code for server:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>

int main()
{
    char *ip = "127.0.0.1";
    int port = 5000;

    int server_sock, client_sock;
    struct sockaddr_in server_addr, client_addr;
```

```
    socklen_t addr_size;
    char buffer[1024];
    int n;

    server_sock = socket(AF_INET, SOCK_STREAM, 0);
    if (server_sock < 0)
    {
        perror("Socket error......\n");

printf("#######################################################\n");
        printf("\n");
        exit(1);
    }

    memset(&server_addr, '\0', sizeof(server_addr));
    server_addr.sin_family = AF_INET;
    server_addr.sin_port = port;
    server_addr.sin_addr.s_addr = inet_addr(ip);

    n = bind(server_sock, (struct sockaddr *)&server_addr,
sizeof(server_addr));
    if (n < 0)
    {
        perror("Bind error......");

printf("#######################################################\n");
        printf("\n");
        exit(1);
    }

    listen(server_sock, 5);
    printf("Serever running......\n");

printf("#######################################################\n");
    printf("\n");

    addr_size = sizeof(client_addr);
    client_sock = accept(server_sock, (struct sockaddr *)&client_addr,
&addr_size);
    printf("Client Connected......\n");

printf("#######################################################\n");
    printf("\n");

    while (1)
    {
        bzero(buffer, 1024);
        recv(client_sock, buffer, sizeof(buffer), 0);
```

```
        printf("Message from client is : %s\n", buffer);
        printf("\n");

        if ((strncmp(buffer, "exit", 4)) == 0)
        {
            // When client send exit request
            close(client_sock);
            printf("Client disconnected......\n");

printf("#######################################################\n");
            printf("\n");

            // For client to enter server again so not stuck here
            client_sock = accept(server_sock, (struct sockaddr
*)&client_addr, &addr_size);
            printf("Client Connected......\n");

printf("#######################################################\n");
            printf("\n");
        }
    }

    return 0;
}
```

Output for client:

```
nihar@nihar: ~/M-Tech/Sem 2/Distributed Systems/LABS/LAB 3
File  Actions  Edit  View  Help
┌──(nihar㉿nihar)-[~/…/Sem 2/Distributed Systems/LABS/LAB 3]
└─$ gcc client.c -o client

┌──(nihar㉿nihar)-[~/…/Sem 2/Distributed Systems/LABS/LAB 3]
└─$ ./client
TCP server socket created......
Server connected......
#######################################################

Enter message to send a server (exit for quit) :
test
Your message sended to server is :
test

Enter message to send a server (exit for quit) :
test2
Your message sended to server is :
test2

Enter message to send a server (exit for quit) :
exit
Your message sended to server is :
exit

Server disconnected......
```

Output for server:

```
nihar@nihar: ~/M-Tech/Sem 2/Distributed Systems/LABS/LAB 3
File  Actions  Edit  View  Help
┌──(nihar㉿nihar)-[~/…/Sem 2/Distributed Systems/LABS/LAB 3]
└─$ gcc server.c -o server

┌──(nihar㉿nihar)-[~/…/Sem 2/Distributed Systems/LABS/LAB 3]
└─$ ./server
Serever running......
#######################################################

Client Connected......
#######################################################

Message from client is : test

Message from client is : test2

Message from client is : exit

Client disconnected......
#######################################################
```