P. ge No.
D te.

LAB ASSIGNMENT-2:

Write a server program and client program for TCP connection.

Server_TCP.c

```
#include <stdio.h>
#include <netdb.h>
#include <netinet/in.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <unistd.h> //read(), write(), close()
#define MAX 80
#define PORT 8080.
#define SA struct sockaddr
  #Function designed for chat b/w client & server
    void func(int connfd)
        {
          char buff[MAX];
          int n;
          // infinite loop for chat
          for(;;) {
            bzero( buff, MAX);
        // Read the message from client & copy it in buffer
```

Teacher's Signature .....

```
        read( connfd, buff, sizeof(buff));
// print buffer which contains the client contents.
    printf(" From Client: %s \t To client: ", buff);
        bzero( buff, MAX);
        n = 0;
        // copy server message in the buffer
    while(( buff[n++] = getchar()) != '\n');

    // and send that buffer to client.
    write ( connfd, buff, sizeof(buff));

    // if msg contains "Exit" the server exit & chat
      ended.
    if (strncmp ("exit", buff, 4) == 0 ){
        printf(" Server Exit ... \n");
        break;
    }
}

    // Driver function.
int main()
    {
    int sockfd, connfd, len;
    struct sockaddr_in servaddr, cli;
    // socket create and verification.
    sockfd = socket ( AF_INET, SOCK_STREAM, 0);
```

Page No.
Date.

```
if ( sockfd == -1) {
    printf(" socket creation failed ... \n");
    exit (0);
}
else
    printf(" Socket successfully created.. \n");
bzero (& servaddr, sizeof( servaddr));

// assign IP, PORT
servaddr. sin_family = AF_INET;
servaddr. sin_addr. s_addr = htonl(INADDR_ANY);
servaddr. sin_port = htons (PORT);

// Binding newly created socket to given IP &
   verification
if ((bind( sockfd, (SA*) & servaddr, sizeof( servaddr))) != 0)
    {
    printf(" socket bind failed --- \n");
    exit (0);
}
else
    printf(" Socket successfully binded -- \n");
// Now server is ready to listen & verification.
    if (( listen( sockfd, 5) ) != 0)
        {
        printf(" Listen failed --- \n");
        exit (0);
    }
```

Teacher's Signature .....

```
    else
        printf("Server listening..\n");
        len = sizeof(cli);
// Accept the data packet from client & verification
    connfd = accept(sockfd, (SA*)&cli, &len);
        if(connfd < 0)
            {
                printf("Server Accept failed...\n");
                exit(0);
            }
    else
        printf("Server accept the client...\n");
// Function for chatting b/w client & server
        func(connfd);
// After chatting close the socket
        close(sockfd);
}
```

client_TCP.c

```
#include<stdio.h>
#include<arpa/inet.h>// inet_addr()
#include<netdb.h>
#include<stdlib.h>
#include<string.h>
#include<strings.h> // bzero()
#include<sys/socket.h>
```

Teacher's Signature .....

Page No.
Date.

```
#include <unistd.h> // read(), write(), close()
#define MAX 80
#define PORT 8080
#define SA struct sockaddr
void func(int sockfd)
{
    char buff[MAX];
    int n;
    for(;;) {
        bzero(buff, sizeof(buff));
        printf("Enter the String: ");
        n = 0;
        while((buff[n++] = getchar()) != '\n');
        write(sockfd, buff, sizeof(buff));
        bzero(buff, sizeof(buff));
        read(sockfd, buff, sizeof(buff));
        printf("From Server: %s", buff);
        if((strncmp(buff, "exit", 4)) == 0) {
            printf("Client Exit...\n");
            break;
        }
    }
}
int main()
{
    int sockfd, connfd;
```

Teacher's Signature .....

```
struct sockaddr_in servaddr, cli;
// socket create & verification.
sockfd = socket(AF_INET, SOCK_STREAM, 0);
if (sockfd == -1) {
    printf("Socket Creation failed ...\n");
    exit(0);
}
else
    printf("Socket Successfully created ...\n");
bzero(&servaddr, sizeof(servaddr));
// assign IP, port;
servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
servaddr.sin_port = htons(PORT);
    //Connect the client socket to server socket.
if (connect(sockfd, (SA*)&servaddr, sizeof(servaddr))
        != 0) {
    printf("connection with the server failed ...\n");
    exit(0);
}
else
    printf("connected to the server ..\n");
// function for chat
    func(sockfd);
// close the socket
    close(sockfd);
}
```

Teacher's Signature .....

Output

```
Socket successfully created..
Socket successfully binded..
Server listening..
Server accept the client...
From Client: Sayanabha.
To Client: Chandra.
From Client: Computer Networks.
To Client: Lab.
From Client: exit
To Client: Server Exit...
```