Sample Program:

AIM:  
           To write a program for TCP echo client server.  
  
ALGORITHM:  
SERVER:  
    STEP 1: Start  
    STEP 2: Declare the variables for the socket  
    STEP 3: Specify the family, protocol, IP address and port number  
    STEP 4: Create a socket using socket() function  
    STEP 5: Bind the IP address and Port number  
    STEP 6: Listen and accept the client’s request for the connection  
    STEP 7: Read the client’s message  
    STEP 8: Display the client’s message  
    STEP 9: Close the socket  
    STEP 10: Stop  
CLIENT:  
    STEP 1: Start  
    STEP 2: Declare the variables for the socket  
    STEP 3:  Specify the family, protocol, IP address and port number  
    STEP 4: Create a socket using socket() function  
    STEP 5: Call the connect() function  
    STEP 6: Read the input message  
    STEP 7: Send the input message to the server  
    STEP 8: Display the server’s echo  
    STEP 9: Close the socket  
    STEP 10: Stop  
  
SOURCE CODE:  
SERVER:  
#include<stdio.h>  
#include<netinet/in.h>  
#include<netdb.h>   
int main()  
{  
       int sockfd,newsockfd,clength,n;  
       struct sockaddr\_in serv\_addr,cli\_addr;  
       char buffer[4096];  
       sockfd=socket(AF\_INET,SOCK\_STREAM,0);  
       serv\_addr.sin\_family=AF\_INET;  
       serv\_addr.sin\_addr.s\_addr=INADDR\_ANY;

printf(“enter port num: “);

scanf(%d”,&n);  
       serv\_addr.sin\_port=htons(n);  
       printf("\nStart");  
       bind(sockfd,(struct sockaddr\*)&serv\_addr,sizeof(serv\_addr));  
       printf("\nListening...");  
       printf("\n");  
       listen(sockfd,5);  
       clength=sizeof(cli\_addr);  
       newsockfd=accept(sockfd,(struct sockaddr\*)&cli\_addr,&clength);  
       printf("\nAccepted");  
       printf("\n");  
       read(newsockfd,buffer,4096);  
       printf("\nClient message:%s",buffer);  
       write(newsockfd,buffer,4096);  
       printf("\n");  
       close(sockfd);  
       return 0;  
}  
CLIENT:  
#include<stdio.h>  
#include<sys/types.h>  
#include<sys/socket.h>  
#include<netinet/in.h>  
#include<netdb.h>  
#define SERV\_TCP\_PORT 5035  
int main()  
{  
       int sockfd,n;  
       struct sockaddr\_in serv\_addr;  
       struct hostent \*server;  
       char buffer[4096];  
       sockfd=socket(AF\_INET,SOCK\_STREAM,0);  
       serv\_addr.sin\_family=AF\_INET;  
      serv\_addr.sin\_addr.s\_addr=INADDR\_ANY;  
       serv\_addr.sin\_port=htons(n);

printf(“enter port num: “);

scanf(%d”,&n);  
       printf("\nReady for sending...");  
       connect(sockfd,(struct sockaddr\*)&serv\_addr,sizeof(serv\_addr));  
       printf("\nEnter the message to send\n");  
       printf("\nClient: ");  
       fgets(buffer,4096,stdin);  
       write(sockfd,buffer,4096);  
       printf("Serverecho:%s",buffer);  
       printf("\n");  
       close(sockfd);  
       return 0;  
}

AIM:  
           To write a program for UDP echo client server.  
  
ALGORITHM:  
SERVER:  
    STEP 1: Start  
    STEP 2: Declare the variables for the socket  
    STEP 3: Specify the family, protocol, IP address and port number  
    STEP 4: Create a socket using socket() function  
    STEP 5: Bind the IP address and Port number  
    STEP 6: Listen and accept the client’s request for the connection  
    STEP 7: Read and Display the client’s message  
    STEP 8: Stop  
CLIENT:  
    STEP 1: Start  
    STEP 2: Declare the variables for the socket  
    STEP 3:  Specify the family, protocol, IP address and port number  
    STEP 4: Create a socket using socket() function  
    STEP 5: Call the connect() function  
    STEP 6: Read the input message  
    STEP 7: Send the input message to the server  
    STEP 8: Display the server’s echo  
    STEP 9: Close the socket  
    STEP 10: Stop  
  
SOURCE CODE:  
SERVER:  
#include<sys/types.h>  
#include<sys/socket.h>  
#include<netinet/in.h>  
#include<unistd.h>  
#include<netdb.h>  
#include<stdio.h>  
#include<string.h>  
#include<arpa/inet.h>  
#define MAXLINE 1024  
int main()  
{  
int sockfd;  
int n;  
socklen\_t len;  
char msg[1024];  
struct sockaddr\_in servaddr,cliaddr;  
sockfd=socket(AF\_INET,SOCK\_DGRAM,0);  
servaddr.sin\_family=AF\_INET;  
servaddr.sin\_addr.s\_addr=INADDR\_ANY;  
servaddr.sin\_port=htons(5035);  
printf("\n\n Binded");  
bind(sockfd,(struct sockaddr\*)&servaddr,sizeof(servaddr));  
printf("\n ");  
len=sizeof(cliaddr);  
n=recvfrom(sockfd,msg,MAXLINE,0,(struct sockaddr\*)&cliaddr,&len);  
printf("\n Client's Message : %s\n",msg);  
sendto(sockfd,msg,n,0,(struct sockaddr\*)&cliaddr,len);  
return 0;  
}  
  
CLIENT:  
#include<sys/types.h>  
#include<sys/socket.h>  
#include<netinet/in.h>  
#include<string.h>  
#include<arpa/inet.h>  
#include<string.h>  
#include<arpa/inet.h>  
#include<stdio.h>  
#define MAXLINE 1024  
int main(int argc,char\* argv[])  
{  
int sockfd;  
int n;  
socklen\_t len;  
char sendline[1024],recvline[1024];  
struct sockaddr\_in servaddr;  
strcpy(sendline,"");  
printf("\n Enter the message : ");  
scanf("%s",sendline);  
sockfd=socket(AF\_INET,SOCK\_DGRAM,0);  
servaddr.sin\_family=AF\_INET;  
servaddr.sin\_addr.s\_addr=inet\_addr("127.0.0.1");  
servaddr.sin\_port=htons(5035);  
len=sizeof(servaddr);  
sendto(sockfd,sendline,MAXLINE,0,(struct sockaddr\*)&servaddr,len);  
n=recvfrom(sockfd,recvline,MAXLINE,0,NULL,NULL);  
recvline[n]=0;  
printf("\n Server's Echo : %s\n\n",recvline);  
return 0;  
}