**1 . Write a program for TCP echo client-server in c.**

**// Server**

//Write a program for TCP echo client-server in c

#include<stdio.h> // basic c header

#include<netinet/in.h> // has the sockaddr\_in structure

#include<unistd.h> // has functions for read and write operations

#define PORT 5035

int main(){

    int sockfd,newsockfd,clength;

    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\_port = htons(PORT);

    printf("\nStart");

    bind(sockfd,(struct sockaddr\*)&serv\_addr,sizeof(serv\_addr));

    printf("\nListening...\n");

    listen(sockfd,5);

    clength = sizeof(cli\_addr);

    newsockfd = accept(sockfd,(struct sockaddr\*)&cli\_addr,&clength);

    printf("\nAccepted\n");

    read(newsockfd,buffer,4096);

    printf("\nClient message: %s",buffer);

    write(newsockfd,buffer,4096);

    close(sockfd);

    return 0;

}

**//Client**

//Write a program for TCP echo client-server in c

#include<stdio.h> // basic c header

#include<unistd.h> // has the sockaddr\_in structure

#include<netinet/in.h> // has functions for read and write operations

#include<arpa/inet.h> //inet\_addr

#define PORT 5035

int main(){

    int sockfd;

    struct sockaddr\_in serv\_addr;

    char buffer[4096];

    sockfd = socket(AF\_INET, SOCK\_STREAM,0);

    serv\_addr.sin\_family = AF\_INET;

    serv\_addr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

    serv\_addr.sin\_port = htons(PORT);

    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("Server echo: %s",buffer);

    close(sockfd);

    return 0;

}

**2 . Write a program for UDP echo client-server in c.**

**// Server**

//Write a program for UDP echo client-server in c

#include<stdio.h>

#include<netinet/in.h> // sockaddr\_in

#include<string.h> // bzero

#define PORT 5035

#define MAXLINE 1024

int main(){

    int sockfd,n;

    socklen\_t len;

    char msg[1024];

    struct sockaddr\_in serv\_addr,cli\_addr;

    sockfd=socket(AF\_INET,SOCK\_DGRAM,0);

    bzero(&serv\_addr,sizeof(serv\_addr));

    serv\_addr.sin\_family=AF\_INET;

    serv\_addr.sin\_addr.s\_addr=INADDR\_ANY;

    serv\_addr.sin\_port=htons(PORT);

    printf("Binded");

    bind(sockfd,(struct sockaddr\*)&serv\_addr,sizeof(serv\_addr));

    printf("\nListening...\n\n");

    for(;;){

        len=sizeof(cli\_addr);

        n=recvfrom(sockfd,msg,MAXLINE,0,(struct sockaddr\*)&cli\_addr,&len);

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

        if(n<6)

            perror("send error");

            sendto(sockfd,msg,n,0,(struct sockaddr\*)&cli\_addr,len);

    }

    return 0;

}

**//Client**

//Write a program for UDP echo client-server in c

#include<stdio.h>

#include<netinet/in.h> // sockaddr\_in

#include<string.h> // bzero

#include<arpa/inet.h> // inet\_addr

#define PORT 5035

#define MAXLINE 1024

int main(){

    int sockfd,n;

    socklen\_t len;

    char sendline[1024],recvline[1024];

    struct sockaddr\_in servaddr;

    strcpy(sendline,"");

    printf("Enter the message: ");

    scanf("%s",sendline);

    sockfd=socket(AF\_INET,SOCK\_DGRAM,0);

    bzero(&servaddr,sizeof(servaddr));

    servaddr.sin\_family=AF\_INET;

    servaddr.sin\_addr.s\_addr=inet\_addr("127.0.0.1");

    servaddr.sin\_port=htons(PORT);

    connect(sockfd,(struct sockaddr\*)&servaddr,sizeof(servaddr));

    len=sizeof(servaddr);

    sendto(sockfd,sendline,MAXLINE,0,(struct sockaddr\*)&servaddr,len);

    n=recvfrom(sockfd,recvline,MAXLINE,0,NULL,NULL);

    recvline[n]=0;

    printf("\nServer's Echo : %s\n",recvline);

    return 0;

}

**3. Write a program for TCP chat client-server in c.**

**// Server**

//Write a program for TCP chat client-server in c

#include<stdio.h>

#include<netinet/in.h> // sockaddr\_in

#include<string.h> // bzero

#include<arpa/inet.h> // inet\_addr

#include<unistd.h> // read , write

#define PORT 5035

int main(){

       int sockfd,newsockfd,clength;

       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;

       serv\_addr.sin\_port=htons(PORT);

       bind(sockfd,(struct sockaddr\*)&serv\_addr,sizeof(serv\_addr));

       listen(sockfd,5);

       clength=sizeof(cli\_addr);

       newsockfd=accept(sockfd,(struct sockaddr\*)&cli\_addr,&clength);

       read(newsockfd,buffer,4096);

       while(buffer!="quit")

       {

         printf("Client message: %s",buffer);

         printf("\nType your message: ");

         fgets(buffer,4096,stdin);

         write(newsockfd,buffer,4096);

         read(newsockfd,buffer,4096);

       }

       close(sockfd);

       return 0;

}

**//Client**

//Write a program for TCP chat client-server in c

#include<stdio.h>

#include<netinet/in.h> // sockaddr\_in

#include<string.h> // bzero

#include<arpa/inet.h> // inet\_addr

#include<unistd.h> // read , write

#define PORT 5035

int main(){

       int sockfd;

       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=inet\_addr("127.0.0.1");

       serv\_addr.sin\_port=htons(PORT);

       connect(sockfd,(struct sockaddr\*)&serv\_addr,sizeof(serv\_addr));

       printf("Enter the message to send : ");

       fgets(buffer,4096,stdin);

       fputs(buffer,stdout);

       while(buffer!="quit")

       {

         if(buffer=="quit")

         break;

         write(sockfd,buffer,4096);

         read(sockfd,buffer,4096);

         printf("\nServer message: %s",buffer);

         printf("\nType your message: ");

         fgets(buffer,4096,stdin);

       }

       close(sockfd);

       return(0);

}

**4 . Write a program for UDP chat client-server in c.**

**// Server**

//Write a program for UDP chat client-server in c

#include<stdio.h>

#include<netinet/in.h>// sockaddr\_in

#include<string.h> // bzero

#include<arpa/inet.h> // inet\_addr

#include<stdlib.h> // exit

#include<unistd.h> // close

#define MAX 80

#define PORT 43454

void func(int sockfd){

    char buff[MAX];

    int n,clen;

    struct sockaddr\_in cli;

    clen=sizeof(cli);

    for(;;){

        bzero(buff,MAX);

        recvfrom(sockfd,buff,sizeof(buff),0,(struct sockaddr\*)&cli,&clen);

        printf("\nFrom client: %s \nTo client: ",buff);

        bzero(buff,MAX);

        n=0;

        while((buff[n++]=getchar())!='\n');

        sendto(sockfd,buff,sizeof(buff),0,(struct sockaddr\*)&cli,clen);

        if(strncmp("exit",buff,4)==0){

            printf("Server Exit...\n");

            break;

        }

    }

}

int main(){

    int sockfd;

    struct sockaddr\_in servaddr;

    sockfd=socket(AF\_INET,SOCK\_DGRAM,0);

    if(sockfd==-1){

        printf("socket creation failed...\n");

        exit(0);

    }

    else

        printf("Socket successfully created..\n");

    bzero(&servaddr,sizeof(servaddr));

    servaddr.sin\_family=AF\_INET;

    servaddr.sin\_addr.s\_addr=htonl(INADDR\_ANY);

    servaddr.sin\_port=htons(PORT);

    if((bind(sockfd,(struct sockaddr\*)&servaddr,sizeof(servaddr)))!=0)

    {

        printf("socket bind failed...\n");

        exit(0);

    }

    else

    printf("Socket successfully binded..\n");

    func(sockfd);

    close(sockfd);

}

**//Client**

//Write a program for UDP chat client-server in c

#include<stdio.h>

#include<netinet/in.h>// sockaddr\_in

#include<string.h> // bzero

#include<arpa/inet.h> // inet\_addr

#include<stdlib.h> // exit

#include<unistd.h> // close

#define MAX 80

#define PORT 43454

int main(){

    char buff[MAX];

    int sockfd,len,n;

    struct sockaddr\_in servaddr;

    sockfd=socket(AF\_INET,SOCK\_DGRAM,0);

    if(sockfd==-1){

        printf("socket creation failed...\n");

        exit(0);

    }

    else

        printf("Socket successfully created..\n");

    bzero(&servaddr,sizeof(len));

    servaddr.sin\_family=AF\_INET;

    servaddr.sin\_addr.s\_addr=inet\_addr("127.0.0.1");

    servaddr.sin\_port=htons(PORT);

    len=sizeof(servaddr);

    for(;;){

        printf("\nEnter string: ");

        n=0;

        while((buff[n++]=getchar())!='\n');

        sendto(sockfd,buff,sizeof(buff),0,(struct sockaddr\*)&servaddr,len);

        bzero(buff,sizeof(buff));

        recvfrom(sockfd,buff,sizeof(buff),0,(struct sockaddr\*)&servaddr,&len);

        printf("From Server: %s\n",buff);

        if(strncmp("exit",buff,4)==0){

            printf("Client Exit...\n");

            break;

        }

    }

    close(sockfd);

}

**5. Write a program for IPC message queue in c**

**// Server**

// Write a program for IPC message queue in c

#include <stdio.h>

#include<stdlib.h> // exit

#include <string.h> // strcpy

#include <sys/msg.h> //  msgsnd, msgrcv

int main()

{

    struct msgbht

    {

        long mtype;

        char mtext[100];

    } send, recv;

    int msg, len;

    msg = msgget((key\_t)25, IPC\_CREAT | 0666);

    if (msg < 0)

    {

        perror("Message faild");

        exit(1);

    }

    strcpy(send.mtext, "Hello, How are you?");

    send.mtype = 1;

    len = strlen(send.mtext);

    if (msgsnd(msg, &send, len, 0) == -1)

    {

        perror("Message faild");

        exit(1);

    }

    if (msgrcv(msg, &recv, 100, 2, 0) == -1)

    {

        perror("Message faild");

        exit(1);

    }

    printf("Message from program 2 is : \n%s\n", recv.mtext);

}

**//Client**

// Write a program for IPC message queue in c

#include <stdio.h>

#include<stdlib.h> // exit

#include <string.h> // strcpy

#include <sys/msg.h> //  msgsnd, msgrcv

int main()

{

    struct msgbht

    {

        long mtype;

        char mtext[100];

    } send, recv;

    int msg, len;

    msg = msgget((key\_t)25, IPC\_CREAT | 0666);

    if (msg < 0)

    {

        perror("Message faild");

        exit(1);

    }

    strcpy(send.mtext, "I am fine thank you");

    send.mtype = 2;

    len = strlen(send.mtext);

    if (msgrcv(msg, &recv, 100, 1, 0) == -1)

    {

        perror("Message faild");

        exit(1);

    }

    if (msgsnd(msg, &send, len, 0) == -1)

    {

        perror("Message faild");

        exit(1);

    }

    printf("Message from program 1 is : \n%s\n", recv.mtext);

}

**6. Write a program for TCP socket (date & time) in c**

**// Server**

// Write a program for TCP socket(date & time) in c

#include<stdio.h>

#include<netinet/in.h> // sockaddr\_in

#include<string.h> // bzero

#include<time.h> // time

#include <unistd.h> //wrire

#include <stdlib.h> //exit

int main(){

    struct sockaddr\_in sa ,cli;

    int sockfd,conntfd,len,ch;

    char str[100];

    time\_t tick;

    sockfd=socket(AF\_INET,SOCK\_STREAM,0);

    if(sockfd<0){

        printf("error in socket\n");

        exit(0);

    }else

        printf("Socket opened\n");

    bzero(&sa,sizeof(sa));

    sa.sin\_port=htons(5600);

    sa.sin\_addr.s\_addr=htonl(0);

    if(bind(sockfd,(struct sockaddr\*)&sa,sizeof(sa))<0){

        printf("Error in binding\n");

    }

    else

        printf("Binded Successfully\n");

    listen(sockfd,50);

    for(;;){

        len=sizeof(ch);

        conntfd=accept(sockfd,(struct sockaddr\*)&cli,&len);

        printf("Accepted\n");

        tick=time(NULL);

        snprintf(str,sizeof(str),"%s",ctime(&tick));

        printf("%s",str);

        write(conntfd,str,100);

    }

}

**//Client**

// Write a program for TCP socket(date & time) in c

#include<stdio.h>

#include<netinet/in.h> // ockaddr\_in

#include <stdlib.h> // exit

#include <strings.h> // bzero

#include <unistd.h> // read

int main(){

    struct sockaddr\_in sa,cli;

    int n,sockfd,len;

    char buff[100];

    sockfd=socket(AF\_INET,SOCK\_STREAM,0);

    if(sockfd<0){

        printf("Error in Socket\n");

        exit(0);

    }

    else

        printf("Socket is Opened\n");

    bzero(&sa,sizeof(sa));

    sa.sin\_family=AF\_INET;

    sa.sin\_port=htons(5600);

    if(connect(sockfd,(struct sockaddr\*)&sa,sizeof(sa))<0){

        printf("Error in connection failed\n");

        exit(0);

    }else

        printf("connected successfully\n");

    if(n=read(sockfd,buff,sizeof(buff))<0){

        printf("Error in Reading\n");

        exit(0);

    }

    else

        printf("Message Read: %s\n",buff);

}

**7. Write a program for socket programming in c**

**// Server**

// Write a program for socket programming in c

#include<stdio.h>

#include<sys/socket.h>

#include<stdlib.h>

#include<netinet/in.h>

#include<string.h>

#include<unistd.h>

#define PORT 8080

int main(){

    int sock\_fd, new\_socket, valread;

    struct sockaddr\_in address;

    int addrlen = sizeof(address);

    char buffer[2048];

    char \*fun = "Souvik Mondal from Server";

    if((sock\_fd = socket(AF\_INET, SOCK\_STREAM, 0)) == 0){

        printf("Error, SCOKET went wrong\n");

        return -1;

    }

    address.sin\_family = AF\_INET;

    address.sin\_addr.s\_addr = INADDR\_ANY;

    address.sin\_port = htons(PORT);

    if(bind(sock\_fd, (struct sockaddr \*)&address,sizeof(address))<0){

        printf("Binding went wrong\n");

        return -1;

    }

    if(listen(sock\_fd, 4)<0){

        printf("Listining went wrong\n");

        return -1;

    }

    if((new\_socket = accept(sock\_fd, (struct sockaddr \*)&address, (socklen\_t\*)&addrlen))<0){

        printf("Accept went wrong\n");

        return -1;

    }

    valread = read(new\_socket, buffer, 2048);

    printf("%s\n", buffer);

    send(new\_socket, fun, strlen(fun),0);

    printf("The message sent\n");

    return 0;

}

**//Client**

// Write a program for socket programming in c

#include<stdio.h>

#include<sys/socket.h>

#include<stdlib.h>

#include<netinet/in.h>

#include<arpa/inet.h>

#include<string.h>

#include<unistd.h>

#define PORT 8080

int main(){

    struct sockaddr\_in address;

    int sock = 0, valread;

    struct sockaddr\_in serv\_addr;

    char \*fun\_client = "Client";

    char buffer[2048] = {0};

    if((sock = socket(AF\_INET, SOCK\_STREAM, 0))<0){

        printf("\nSocket went wrong\n");

        return -1;

    }

    memset(&serv\_addr, '0', sizeof(serv\_addr));

    serv\_addr.sin\_family = AF\_INET;

    serv\_addr.sin\_port = htons(PORT);

    if(inet\_pton(AF\_INET, "127.0.0.1", &serv\_addr.sin\_addr)<=0){

        printf("\nInvalid Address");

        return -1;

    }

    if(connect(sock, (struct sockaddr \*)&serv\_addr, sizeof(serv\_addr))<0){

        printf("No Connection");

        return -1;

    }

    send(sock, fun\_client, strlen(fun\_client), 0);

    printf("Hello message sent\n");

    printf("%s\n", buffer);

    return 0;

}

**8. Write a program to send four successive messages from the sender in c**

**// Server**

// Write a program to send four successive messages from the sender in c

#include<sys/ipc.h>

#include<sys/msg.h>

#include<string.h>

#include<stdlib.h>

#include <stdio.h>

int main(){

    struct msgbht{

        long mtype;

        char mtext[100];

    }send,recv;

    int msg,len;

    msg=msgget((key\_t)25,IPC\_CREAT|0666);

    if(msg<0){

        perror("Message failed\n");

        exit(1);

    }

    strcpy(send.mtext,"Hello, How are you?\n");

    send.mtype=1;

    len=strlen(send.mtext);

    if(msgsnd(msg,&send,len,0)==-1){

        perror("Message failed\n");

        exit(1);

    }

    strcpy(send.mtext,"Msg2:Where are you?\n");

    send.mtype=2;

    len=strlen(send.mtext);

    if(msgsnd(msg,&send,len,0)==-1){

        perror("Message failed");

        exit(1);

    }

    strcpy(send.mtext,"Msg3:What are you doing?\n");

    send.mtype=3;

    len=strlen(send.mtext);

    if(msgsnd(msg,&send,len,0)==-1){

        perror("Message failed\n");

        exit(1);

    }

    strcpy(send.mtext,"Msg4:Why are you here now?\n");

    send.mtype=4;

    len=strlen(send.mtext);

    if(msgsnd(msg,&send,len,0)==-1){

        perror("Message failed\n");

        exit(1);

    }

}

**//Client**

// Write a program to send four successive messages from the sender in c

#include<sys/types.h>

#include<sys/ipc.h>

#include<sys/msg.h>

#include<string.h>

#include<stdlib.h>

#include<stdio.h>

int main(){

    struct msgbht{

        long mtype;

        char mtext[100];

    }recv;

    int i;

    int qid=msgget((key\_t)25,IPC\_CREAT|0666);

    if(qid<0){

        perror("Message failed\n");

        exit(1);

    }

    for(i=0;i<4;i++){

        if(msgrcv(qid,&recv,100,0,0)==-1){

            printf("Msg %d(i+1)failed",i);

            exit(1);

        }

        printf("%s",recv.mtext);

    }

}