Server1

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <poll.h>
#include <pthread.h>
#include <signal.h>
#include <sys/select.h>
#include <sys/socket.h>
#include <sys/ipc.h>
#include <sys/un.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <stddef.h>
#include <fcntl.h>
#include <netinet/ether.h>
#include <netinet/ip.h>
#include <netinet/tcp.h>
#include <netinet/udp.h>
#include <netinet/ip_icmp.h>

int nsfd[255],ind = 0;

void* func1(void* arg)
{
   int fd = *(int*)arg,sz;
   char buffer[50];
   while(1)
   {
       if((sz = recv(fd,buffer,50,0))<0)
           perror("Could not read");
       else
       {
           if(sz==0)
               pthread_exit(0);
           buffer[sz] = '\0';
```

```
            for(int i=0;i<sz;i++)
            {
                if(buffer[i]>='a'&&buffer[i]<='z')
                    buffer[i] = buffer[i]-'a'+'A';
            }
            if(send(fd,buffer,sz,0)<0)
                perror("Could not send");
        }
    }
}

void* func2(void* arg)
{
    int fd = *(int*)arg,sz;
    char buffer[50];
    while(1)
    {
        if((sz = recv(fd,buffer,50,0))<0)
            perror("Could not read");
        else
        {
            if(sz==0)
                pthread_exit(0);
            buffer[sz] = '\0';
            for(int i=0;i<sz;i++)
            {
                if(buffer[i]>='A'&&buffer[i]<='Z')
                    buffer[i] = buffer[i]-'A'+'a';
            }
            if(send(fd,buffer,sz,0)<0)
                perror("Could not send");
        }
    }
}

void* func3(void* arg)
{
    int fd = *(int*)arg,sz;
    char buffer[50];
    while(1)
    {
        if((sz = recv(fd,buffer,50,0))<0)
            perror("Could not read");
        else
        {
            if(sz==0)
                pthread_exit(0);
            buffer[sz] = '\0';
            for(int i=0;i<sz;i++)
            {
                if(buffer[i]>='a'&&buffer[i]<='z')
                    buffer[i] = buffer[i]-'a'+'1';
            }
            if(send(fd,buffer,sz,0)<0)
                perror("Could not send");
        }
    }
}

char reply[100];
fd_set readset;

int main(int argc, char const *argv[])
{
    if(argc<4)
    {
        printf("Usage: %s [SERVER NUMBER] [STARTING PORT NUMBER]
[ENDING PORT NUMBER]\n",argv[0]);exit(0);
    }
    int port_start = atoi(argv[2]),port_end = atoi(argv[3]);
    int n = port_end-port_start+1,j = 0,temp = 1;
    int sfd[n];struct sockaddr_in addr[n];
    FD_ZERO(&readset);
    for(int i=port_start;i<=port_end;i++)
    {
        sfd[j] = socket(AF_INET,SOCK_STREAM,0);
        if(sfd[j]<0)
        {
            perror("Could not create socket");continue;
        }

setsockopt(sfd[j],SOL_SOCKET,SO_REUSEADDR|SO_REUSEPORT,&temp,sizeof(t
emp));
        addr[j].sin_family = AF_INET;
        addr[j].sin_addr.s_addr = htonl(INADDR_LOOPBACK);
        addr[j].sin_port = htons(i);
        if(bind(sfd[j],(struct sockaddr*)&addr[j],sizeof(addr[j]))<0)
        {
            perror("Could not bind1");
        }
        else if(listen(sfd[j],10)<0)
        {
            perror("Could not listen");
        }
        else
        {
            FD_SET(sfd[j],&readset);j++;
        }

    }


    int rsfd = socket(AF_INET,SOCK_RAW,253),r,optval = 1;
    setsockopt(rsfd, IPPROTO_IP, SO_BROADCAST, &optval, sizeof(int));
    struct sockaddr_in rawaddr,cl_addr;socklen_t len =
sizeof(cl_addr);
    memset(&rawaddr,0,sizeof(rawaddr));
    rawaddr.sin_family = AF_INET;
    rawaddr.sin_addr.s_addr = inet_addr("127.0.0.2");
    if(bind(rsfd,(struct sockaddr*)&rawaddr,sizeof(rawaddr))<0)
    {
        perror("Could not bind");
    }
    else
    {
        FD_SET(rsfd,&readset);
    }
    char buffer[100];
    while(1)
    {
        r = select(FD_SETSIZE+1,&readset,NULL,NULL,NULL);
        if(r>0)
        {
            if(FD_ISSET(rsfd,&readset))
            {
                if(recvfrom(rsfd,buffer,100,0,(struct
sockaddr*)&cl_addr,&len)<0)
                {
                    perror("Could not receive");
                }
                struct iphdr *ip;
                ip = (struct iphdr*)buffer;char ad[INET_ADDRSTRLEN];
                printf("Remote IP:
%s\n",inet_ntop(AF_INET,&ip->saddr,ad,INET_ADDRSTRLEN));
                sprintf(reply,"Ports available in Server-%s : %s -
%s",argv[1],argv[2],argv[3]);
                if(sendto(rsfd,reply,strlen(reply)+1,0,(struct
sockaddr*)&cl_addr,sizeof(cl_addr))<0)
                {
                    perror("Could not send");
                }
            }
            else
                FD_SET(rsfd,&readset);
            for(int i=0;i<n;i++)
            {
                if(FD_ISSET(sfd[i],&readset))
                {
                    nsfd[ind] = accept(sfd[i],NULL,NULL);
                    if(nsfd[ind]<0)
                    {
                        perror("Could not accept");continue;
                    }
                    pthread_t p;
                    if(argv[1][0]=='1')
                    {
                        pthread_create(&p,NULL,func1,&nsfd[ind]);
                    }
                    else if(argv[1][0]=='2')
                    {
                        pthread_create(&p,NULL,func2,&nsfd[ind]);
                    }
                    else
                        pthread_create(&p,NULL,func3,&nsfd[ind]);
                    ind++;
                }
                else
                    FD_SET(sfd[i],&readset);
            }
        }
    }
    return 0;
}
```

Client1

```
#include "../cn.h"
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <netinet/ip.h>
#include <pthread.h>

int main(int argc, char const *argv[])
{
    if(argc<2)
    {
        printf("Usage: %s [NO_OF_SERVERS_AVAILABLE]
[RAW_IP_ADDRESS]\n",argv[0]);exit(0);
    }
    int rsfd = socket(AF_INET,SOCK_RAW,253),sz,optval = 1;
    setsockopt(rsfd, IPPROTO_IP, SO_BROADCAST, &optval, sizeof(int));
    struct sockaddr_in
rawaddr,cl_addr;memset(&rawaddr,0,sizeof(rawaddr));
    rawaddr.sin_family = AF_INET;
    rawaddr.sin_addr.s_addr = inet_addr("127.0.0.2");
    cl_addr.sin_family = AF_INET;
    cl_addr.sin_addr.s_addr = inet_addr("127.0.0.3");
    if(bind(rsfd,(struct sockaddr*)&cl_addr,sizeof(cl_addr))<0)
    perror("Could not bind");
    else
    printf("Success..\n");
    /*if(connect(rsfd,(struct sockaddr*)&rawaddr,sizeof(rawaddr))<0)
    {
        perror("Could not connect");exit(0);
    }*/
    int n = atoi(argv[1]);
    char buffer[100];
    strcpy(buffer,"?");
    if(sendto(rsfd,buffer,strlen(buffer)+1,0,(struct
sockaddr*)&rawaddr,sizeof(rawaddr))<0)
    {
        perror("Could not send");
    }
    else
    {
        for(int i=0;i<n;i++)
        {
            if(recvfrom(rsfd,buffer,100,0,NULL,NULL)<0){
                perror("Could not read");
            }
            else
            {
                struct iphdr *ip;
                ip = (struct iphdr*)buffer;
                printf("Reading: %s\n",buffer+(ip->ihl*4));
            }
        }
        printf("Enter the port no you want to connect to\n");
        int portno;
        scanf("%d",&portno);while(getchar()!='\n');
        int sfd = socket(AF_INET,SOCK_STREAM,0);
        struct sockaddr_in addr;
        addr.sin_family = AF_INET;
        addr.sin_addr.s_addr = htonl(INADDR_LOOPBACK);
        addr.sin_port = htons(portno);
        if(connect(sfd,(struct sockaddr*)&addr,sizeof(addr))<0)
        {
            perror("Could not connect");
        }
        else
        {
            while(1)
            {
                printf("Enter a string\n");
                scanf("%[^\n]s",buffer);
                while(getchar()!='\n');
                send(sfd,buffer,strlen(buffer),0);
                sz = recv(sfd,buffer,100,0);
                buffer[sz] = '\0';
                printf("Reading: %s\n",buffer);
            }
        }
    }
    return 0;
}
```