#include <stdio.h>

#include <stdlib.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

int main()

{

    int network\_socket;

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

    struct sockaddr\_in server\_address;

    server\_address.sin\_family=AF\_INET;

    server\_address.sin\_port=htons(64000);

    server\_address.sin\_addr.s\_addr=INADDR\_ANY;

    int connection\_status=connect(network\_socket,(struct sockaddr\*)&server\_address,sizeof(server\_address));

*//check error in connection*

    if(connection\_status==-1)

    {

        printf("\n Error in connection");

    }

*//receive data from server*

    char server\_response[256];

    recv(network\_socket,&server\_response,sizeof(server\_response),0);

*//print server respons*

    printf("\n The server sent : %s",server\_response);

*//close socket*

    close(network\_socket);

    return 0;

}

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#define PORT 64000

int main(void)

{

    fd\_set master;

    fd\_set read\_fds;

    struct sockaddr\_in serveraddr;

    struct sockaddr\_in clientaddr;

    int fdmax;

    int listener;

    int newfd;

    char buf[128];

    int nbytes;

    int allow=1;

    socklen\_t addrlen;

    int i,j;

    FD\_ZERO(&master);

    FD\_ZERO(&read\_fds);

    if((listener=socket(PF\_INET,SOCK\_STREAM,0))==-1)

    {

        perror("Socket");

        exit(1);

    }

    if(setsockopt(listener,SOL\_SOCKET,SO\_REUSEADDR,&allow,sizeof(int))==-1)

    {

        perror("setsockopt");

        exit(1);

    }

    serveraddr.sin\_family=AF\_INET;

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

    serveraddr.sin\_port=htons(PORT);

    memset(&(serveraddr.sin\_zero),'\0',8);

    if(bind(listener,(struct sockaddr \*)&serveraddr,sizeof(serveraddr))==-1)

    {

        perror("Bind");

        exit(1);

    }

    printf("\n server is bound at port %d",PORT);

    if(listen(listener,5)==-1)

    {

        perror("listen");

        exit(1);

    }

    printf("\n server started listening at port %d",PORT);

    FD\_SET(listener,&master);

    fdmax=listener;

    while(1)

    {

        read\_fds=master;

        if(select(fdmax+1,&read\_fds,NULL,NULL,NULL)==-1)

        {

            perror("Select");

            exit(1);

        }

        for(i=0;i<=fdmax;i++)

        {

            if(FD\_ISSET(i,&read\_fds))

            {

                if(i==listener)

                {

                    addrlen=sizeof(clientaddr);

                    if((newfd=accept(listener,(struct sockaddr\*)&clientaddr,&addrlen))==-1)

                    {

                        perror("accept");

                    }

                    else{

                        FD\_SET(newfd,&master);

                        if(newfd>fdmax)

                        {

                            fdmax=newfd;

                        }

                        printf("\n server select : new connection from %s on socket %d\n",inet\_ntoa(clientaddr.sin\_addr),newfd);

                        char welcome[50]="Welcome from server Umang kumar\n";

                        send(fdmax,welcome,sizeof(welcome),0);

                    }

                }

                else{

                    memset(buf,0,sizeof(buf));

                    if((nbytes=recv(i,buf,sizeof(buf),0))<=0)

                    {

                        if(nbytes==0)

                        {

                            printf("\n select server : socket %d hung up \n",i);

                        }

                        else{

                            perror("recv");

                        }

                        close(i);

                        FD\_CLR(i,&master);

                    }

                    else{

                        for(j=0;j<=fdmax;j++)

            {

                            if(FD\_ISSET(j,&master))

                            {

                                if(j!=listener && j!=i)

                                {

                                    if(send(j,buf,nbytes,0)==-1)

                                    {

                                        perror("send");

                                    }

                                }

                            }

                        }

                    }

                }

}

                }

            }

    return 0;

}

