Mserver.c

#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>

#include<netdb.h>

#define PORT 4950

#define BUFSIZE 1024

void send\_to\_all(int j,int i,int sockfd,int nbytes\_recvd,char \*recv\_buf,fd\_set \*master)

{

if(FD\_ISSET(j,master))

{

if(j!=sockfd && j!=1)

{

if(send(j,recv\_buf,nbytes\_recvd,0)==-1)

{

perror("send");

}

}

}

}

void send\_recv(int i,fd\_set \*master,int sockfd,int fdmax)

{

int nbytes\_recvd,j;

char recv\_buf[BUFSIZE],buf[BUFSIZE];

if((nbytes\_recvd=recv(i,recv\_buf,BUFSIZE,0))<=0)

{

if(nbytes\_recvd==0)

{

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

}

else

{

perror("recv");

}

close(i);

FD\_CLR(i,master);

}

else

{

recv\_buf[nbytes\_recvd]='\0';

printf("%s\n",recv\_buf);

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

{

send\_to\_all(j,i,sockfd,nbytes\_recvd,recv\_buf,master);

}

}

}

void connection\_accept(fd\_set \*master,int \*fdmax,int sockfd,struct sockaddr\_in \*client\_addr)

{

socklen\_t addrlen;

int newsockfd;

addrlen=sizeof(struct sockaddr\_in);

if((newsockfd=accept(sockfd,(struct sockaddr \*)client\_addr,&addrlen))==-1)

{

perror("accept");

exit(1);

}

else

{

FD\_SET(newsockfd,master);

if(newsockfd>\*fdmax)

{

\*fdmax=newsockfd;

}

printf("new connection from %s on port %d\n",inet\_ntoa(client\_addr->sin\_addr),ntohs(client\_addr->sin\_port));

}

}

void connect\_request(int \*sockfd,struct sockaddr\_in \*my\_addr)

{

int yes=1;

if((\*sockfd=socket(AF\_INET,SOCK\_STREAM,0))==-1)

{

perror("Socket");

exit(1);

}

my\_addr->sin\_family=AF\_INET;

my\_addr->sin\_port=htons(4950);

my\_addr->sin\_addr.s\_addr=INADDR\_ANY;

memset(my\_addr->sin\_zero,'\0',sizeof my\_addr->sin\_zero);

//code to reuse the port wait for 1 minute and code is not required.

if(setsockopt(\*sockfd,SOL\_SOCKET,SO\_REUSEADDR,&yes,sizeof(int))==-1)

{

perror("setsockopt");

exit(1);

}

if(bind(\*sockfd,(struct sockaddr \*)my\_addr,sizeof(struct sockaddr))==-1)

{

perror("Unable to bind");

exit(1);

}

if(listen(\*sockfd,10)==-1)

{

perror("listen");

exit(1);

}

printf("\nTCPServer Waiting for client on port 4950\n");

fflush(stdout);

}

int main()

{

fd\_set master;

fd\_set read\_fds;

int fdmax,i;

int sockfd=0;

struct sockaddr\_in my\_addr,client\_addr;

FD\_ZERO(&master);

FD\_ZERO(&read\_fds);

connect\_request(&sockfd,&my\_addr);

FD\_SET(sockfd,&master);

fdmax=sockfd;

printf("Fdmax before while=%d\n",fdmax);

while(1)

{

read\_fds=master;

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

{

perror("select");

exit(4);

}

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

{

printf("Fdmax inside for=%d\n",fdmax);

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

{

if(i==sockfd)

connection\_accept(&master,&fdmax,sockfd,&client\_addr);

else

send\_recv(i,&master,sockfd,fdmax);

}

}

}

return 0;

}

Mclient.c:

#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>

#include<netdb.h>

#include<errno.h>

#define BUFSIZE 1024

void send\_recv(int i,int sockfd)

{

char send\_buf[BUFSIZE];

char recv\_buf[BUFSIZE];

int nbyte\_recvd,len;

if(i==0)

{

fgets(send\_buf,BUFSIZE,stdin);

if(strcmp(send\_buf,"quit\n")==0)

{

exit(0);

}

else

len=strlen(send\_buf);

send\_buf[len-1]='\0';

send(sockfd,send\_buf,strlen(send\_buf),0);

}

else

{

nbyte\_recvd=recv(sockfd,recv\_buf,BUFSIZE,0);

recv\_buf[nbyte\_recvd]='\0';

printf("%s\n",recv\_buf);

fflush(stdout);

}

}

void connect\_request(int \*sockfd,struct sockaddr\_in \*server\_addr)

{

if((\*sockfd=socket(AF\_INET,SOCK\_STREAM,0))==-1)

{

perror("Socket");

exit(1);

}

server\_addr->sin\_family=AF\_INET;

server\_addr->sin\_port=htons(4950);

server\_addr->sin\_addr.s\_addr=inet\_addr("127.0.0.1");

memset(server\_addr->sin\_zero,'\0',sizeof server\_addr->sin\_zero);

if(connect(\*sockfd,(struct sockaddr\*)server\_addr,sizeof(struct sockaddr))==-1)

{

perror("connect");

exit(1);

}

}

int main()

{

int sockfd,fdmax,i;

struct sockaddr\_in server\_addr;

fd\_set master;

fd\_set read\_fds;

connect\_request(&sockfd,&server\_addr);

FD\_ZERO(&master);

FD\_ZERO(&read\_fds);

FD\_SET(0,&master);

FD\_SET(sockfd,&master);

fdmax=sockfd;

while(1)

{

read\_fds=master;

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

{

perror("select");

exit(4);

}

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

{

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

send\_recv(i,sockfd);

}

}

printf("client-quited\n");

close(sockfd);

return 0;

}









