**//server for file transfer:fileserver.cc**

#include<stdio.h>

#include<unistd.h>

#include<string.h>

**#include<sys/socket.h>**

**#include<netinet/in.h>**

**#include<sys/types.h>**

**#define SERV\_PORT 5576**

main(int argc,char \*\*argv)

{

int i,j;

ssize\_t n;

FILE \*fp;

char s[80],f[80];

**struct sockaddr\_in servaddr,cliaddr;**

int listenfd,connfd,clilen;

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

**bzero(&servaddr,sizeof(servaddr));**

**servaddr.sin\_family=AF\_INET;**

**servaddr.sin\_port=htons(SERV\_PORT);**

**bind(listenfd,(struct sockaddr \*)&servaddr,sizeof(servaddr));**

**listen(listenfd,1);**

**clilen=sizeof(cliaddr);**

**connfd=accept(listenfd,(struct sockaddr\*)&cliaddr,&clilen);**

printf("\n clinet connected");

**read(connfd,f,80);**

fp=fopen(f,"r");

printf("\n name of the file: %s",f);

while(fgets(s,80,fp)!=NULL)

{

printf("%s",s);

**write(connfd,s,sizeof(s));**

}

close(listenfd);

fclose(fp); } //close main

**//file transfer client program:fileclient.cc**

#include<stdio.h>

#include<unistd.h>

#include<string.h>

**#include<sys/socket.h>**

**#include<netinet/in.h>**

**#include<sys/types.h>**

**#include<arpa/inet.h>**

**#define SERV\_PORT 5576**

int main()

{

int i,j;

ssize\_t n;

char filename[80],recvline[80];

**struct sockaddr\_in servaddr;**

int sockfd;

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

**bzero(&servaddr,sizeof(servaddr));**

**servaddr.sin\_family=AF\_INET;**

**servaddr.sin\_port=htons(SERV\_PORT);**

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

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

printf("enter the file name");

scanf("%s",filename);

**write(sockfd,filename,sizeof(filename));**

printf("\n data from server: \n");

while(read(sockfd,recvline,80)!=0)

{

fputs(recvline,stdout);

}

close(sockfd);

}

Open Two different terminals.

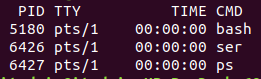
One for client one for server

|  |
| --- |
| Server side |
| Compilation of server  $gcc **fileserver.cc –o ser**  **Execution of server**  **$./ser** |

When the server and client is not closed properly, segmentation fault may come.

Then use ps command to check all running processes

**$ps**

****

**Now kill the ser process**

**$kill -9 6426**

**here 6426 is PID of the ser process.**

|  |
| --- |
| Client Side |
| Compilation of client  $gcc **fileclient.cc –o cli**  **Execution of client**  **$./cli** |

**Todays Assignment:**

Write a socket program, where the client sends content of file and server sends the number of characters in the file.