**Implement a FILE SERVER using PIPE: Write a client-server program using pipe as IPC mechanism in which client passes a filename to the server, server retrieves the contents of the file and passes them to client, and client displays them on the monitor.**

#include<fcntl.h>

void client(int readfd, int writefd){

char buff[1024];

int len,n;

len=read(0,buff,1024);

len--;

write(writefd,buff,len);

while((n=read(readfd,buff,1024))>0)

write(1,buff,n);

}

void server(int readfd, int writefd){

int n,fd;

char buff[1024];

n=read(readfd,buff,1024);

buff[n]=0;

fd=open(buff,O\_RDONLY);

while((n=read(fd,buff,1024))>0)

write(writefd,buff,n);

}

int main(){

int id;

int pipe1[2];

int pipe2[2];

pipe(pipe1);

pipe(pipe2);

id=fork();

if(id==0){

close(pipe1[1]);

close(pipe2[0]);

server(pipe1[0],pipe2[1]);

}

else{

close(pipe2[1]);

close(pipe1[0]);

client(pipe2[0],pipe1[1]);

}

}

Write a program to implement ls | sort | wc.

main(){

int pipe1[2],pipe2[2],id1,id2;

pipe(pipe1);

id1=fork();

if(id1==0){

pipe(pipe2);

id2=fork();

if(id2==0){

close(1);

dup(pipe2[1]);

close(pipe1[0]);

close(pipe1[1]);

close(pipe2[0]);

close(pipe2[1]);

execl("/bin/ls","ls",0);

}

else

{

close(0);

dup(pipe2[0]);

close(1);

dup(pipe1[1]);

close(pipe1[0]);

close(pipe1[1]);

close(pipe2[0]);

close(pipe2[1]);

execl("/bin/sort","sort",0);

}

}

else{

close(0);

dup(pipe1[0]);

close(pipe1[0]);

close(pipe1[1]);

execlp("wc","wc",0);

}

}