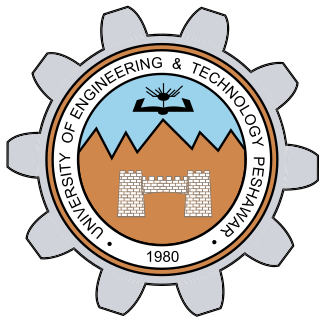
**Assignment No. 03**



**Fall 2023**

**CSE-302 Systems Programming**

Submitted by: **Maaz Habib**

Registration No.: **20PWCSE1952**

Class Section: **C**

“On my honor, as a student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work”

Submitted to:

**Engr. Madiha Sher**

Monday, January 8th, 2023

**Department of Computer Systems Engineering**

**University of Engineering and Technology, Peshawar**

**Task 1:**

**Implement a client-server model to design the “File Server” Server code:**

#include<unistd.h>

#include<error.h>

#include<stdio.h>

#include<stdlib.h>

#include<sys/select.h>

#include<fcntl.h>

#include<sys/stat.h>

#include<dirent.h>

#include<pwd.h>

#include<errno.h>

#include<string.h>

#define size 1000 int fd;

void findcmd(char\* subDir){

DIR \*sub\_dp = opendir(subDir); if (sub\_dp==NULL){

perror("Error is: ");

}

struct dirent \*subDirp; struct stat buf; if(sub\_dp != NULL){

while((subDirp = readdir(sub\_dp)) != NULL){

if(strcmp(subDirp -> d\_name, ".") != 0 && strcmp(subDirp -> d\_name, "..")

!= 0){

write(fd,subDirp->d\_name,strlen(subDirp->d\_name)); write(fd,"\n",1);

}

}

closedir(sub\_dp);

}

else{

printf("%s\n",subDir);

}

}

int main(int argc, char\* agrv[]){ struct timeval timeout; timeout.tv\_sec = 25; timeout.tv\_usec = 0;

char buff1[size]; for(int i=0;i<size;i++){ buff1[i] = '\0';

}

int r = mkfifo("cfifo",S\_IRUSR|S\_IWUSR); if ( r == -1 && errno != EEXIST ) { perror("Faild to Create fifo");

return -1;

}

int bw, br; fd = open("cfifo",O\_RDWR); if ( fd == -1 ) { perror("Faild to open fifo"); return -1;

}

fd\_set readset;

FD\_ZERO(&readset); int max\_fd = fd; while(1){

while(1){ char buff[size]; for(int i=0;i<size;i++){ buff[i] = '\0';

}

FD\_SET(fd,&readset);

int nfds = select(fd + 1,&readset,NULL,NULL,&timeout); if(nfds==-1){

perror("Select error\n"); return -1; }else if(nfds==0){

printf("No file is ready\n"); return -1; }else if(nfds>0){ if(FD\_ISSET(fd,&readset)){ br = read(fd,buff,size);

DIR \*subDir = opendir(buff);

if(subDir != NULL)

{

printf("\nListing Directory\n\n"); sleep(5); //for pause of 5 seconds. findcmd(buff);

chdir(buff);

sleep(2);

}else {

printf("\nSending File\n\n"); sleep(5); //for pause of 5 seconds. int fd2 = open(buff,O\_RDONLY); if ( fd2 == -1 ) { perror("Faild to open fifo"); return -1;

}

br = read(fd2,buff1,size); bw = write(fd,buff1,strlen(buff1)); sleep(5); break;

}

}

}

}

}

return 0;

}

**Client code:**

#include<unistd.h>

#include<error.h>

#include<stdio.h>

#include<stdlib.h>

#include<sys/select.h>

#include<fcntl.h>

#include<sys/stat.h>

#include<dirent.h>

#include<pwd.h>

#include<errno.h>

#include<string.h>

#define size 1000 int main(int argc, char\* argv[]){

char buff1[size]; for(int i=0;i<size;i++){ buff1[i] = '\0';

}

int bw, br; int fd = open("cfifo",O\_RDWR);

if ( fd == -1 ) {

perror("Faild to open fifo");

return -1;

}

br = read(STDIN\_FILENO,buff1,size); bw = write(fd,buff1,strlen(buff1)-1); sleep(6); br = read(fd,buff1,size); printf("%s\n",buff1); while(1){ char buff[size]; for(int i=0;i<size;i++){ buff[i] = '\0';

}

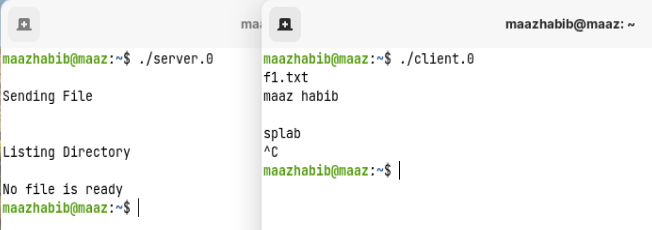
br = read(STDIN\_FILENO,buff,size); bw = write(fd,buff,strlen(buff)-1); sleep(6);

br = read(fd,buff,size); printf("%s\n",buff);

}

}

**Output:**

****

**Task 2:**

**Implement parallel version of “Find” command where different processes search for the file in different directories.**

**Code:**

#include<stdlib.h>

#include<unistd.h>

#include<stdio.h>

#include<sys/stat.h>

#include<dirent.h>

#include<error.h>

#include <libgen.h>

#include <string.h>

void findcmd(char\* subDir){ DIR \*sub\_dp = opendir(subDir); struct dirent \*subDirp; struct stat buf; if(sub\_dp != NULL){

while((subDirp = readdir(sub\_dp)) != NULL){ if(strcmp(subDirp -> d\_name, ".") != 0 && strcmp(subDirp -> d\_name, "..") != 0){ char temp\_full\_path[strlen(subDir) + strlen(subDirp -> d\_name) + 2]; // For giving optimal space.

sprintf(temp\_full\_path,"%s/%s",subDir,subDirp -> d\_name); //String Concatination

printf("%s\n", temp\_full\_path); DIR \*subsubDP = opendir(temp\_full\_path); if(subsubDP != NULL )

{

int x = fork();

closedir(subsubDP); if (x == 0){ findcmd(temp\_full\_path); printf("i am child :%d, and my parrent is: %d\n",getpid(),getppid());

}

}

}

}

closedir(sub\_dp);

}

else{

printf("%s\n",subDir);

}

}

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

{

//printf("you are searching %s\n",argv[1]); findcmd(argv[1]); sleep(1); return 0;

}

**Output:**

