**UNIX I/O (continued)**

**Lab no# 06**

****

**Fall 2021**

**CSE-302 System Programming Lab**

Submitted by: **Ashfaq Ahmad**

Registration No: **19PWCSE1795**

Class Section: **B**

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

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Dr: Ma’am Madeha sheer**

**January** 23, 2022

**Department of Computer Systems Engineering**

**University of Engineering and Technology, Peshawar**

**Parallel file copying or multiple Reads/Writes and Cat utility**

**Lab Objective(s):**

* Understand and implement read, write, open, close functions.

**Task # 01:** Write a program for parallel file copying using multiple processes.

**Code:**

#include <stdio.h>

#include <unistd.h>

#include <stdlib.h>

#include <fcntl.h>

#include <sys/stat.h>

#include <sys/wait.h>

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

{

int x;

if(argc<3 && argc%2==0) //user arguments must be even + 1 this program itself=odd. so argc%2 will no equal to 0.

//user input arguments must be at least 2 + this program=3. so argc>3.

{

printf("Invalid no of arguments:\n");

return -1;

}

for (int i=1; i<argc; i+=2) //this loop will iterate argc-1/2 time. odd numbr files opened for reading.

{

x=fork(); // "argc-1/2" child will be created. each child will used two files one for read other for write

//called parallel file copying.

if(x==-1)

{

perror("Error! Child can't created");

return -1;

}

if(x==0) //fan condition

{

int fd=open(argv[i],O\_RDONLY);

if(fd==-1)

{

printf("Sorry! file %s can't opened successfully:\n",argv[i]);

perror("Reason");

return -1;

}

//even numbr no files opened for writing.

Int fdc=open(argv[i+1],O\_WRONLY|O\_CREAT|O\_TRUNC,S\_IRWXU,S\_IRWXG,S\_IRWXO);

if(fdc==-1)

{

printf("Sorry! file %s can't opened successfully:\n",argv[i+1]);

perror("Reason");

return -1;

}

char buffer[100];

int bytesread;

do

{

bytesread=read(fd,buffer,sizeof(buffer));

if(bytesread==-1)

{

printf("Sorry! Data Can't read from %s: ",argv[i]);

perror("Reason");

return -1;

}

int byteswrite=write(fdc,buffer,bytesread);

if(byteswrite==-1)

{

printf("Sorry! Data Can't write on %s: ",argv[i+1]);

perror("Reason");

return -1;

}

}while(bytesread!=0);

int fd1=close(fd);

if(fd1==-1)

{

printf("%s can't closed successfully:\n",argv[i]);

perror("Reason");

return -1;

}

int fd2=close(fdc);

if(fd2==-1)

{

printf("%s can't closed successfully:\n",argv[i+1]);

perror("Reason");

return -1;

}

}

}

if(x>0)

{

for(int i=1; i<argc/2; i++) //as argc-1/2 child created so parent process will wait equal times.

{

int x=wait(NULL); //each child don'treturn anything.

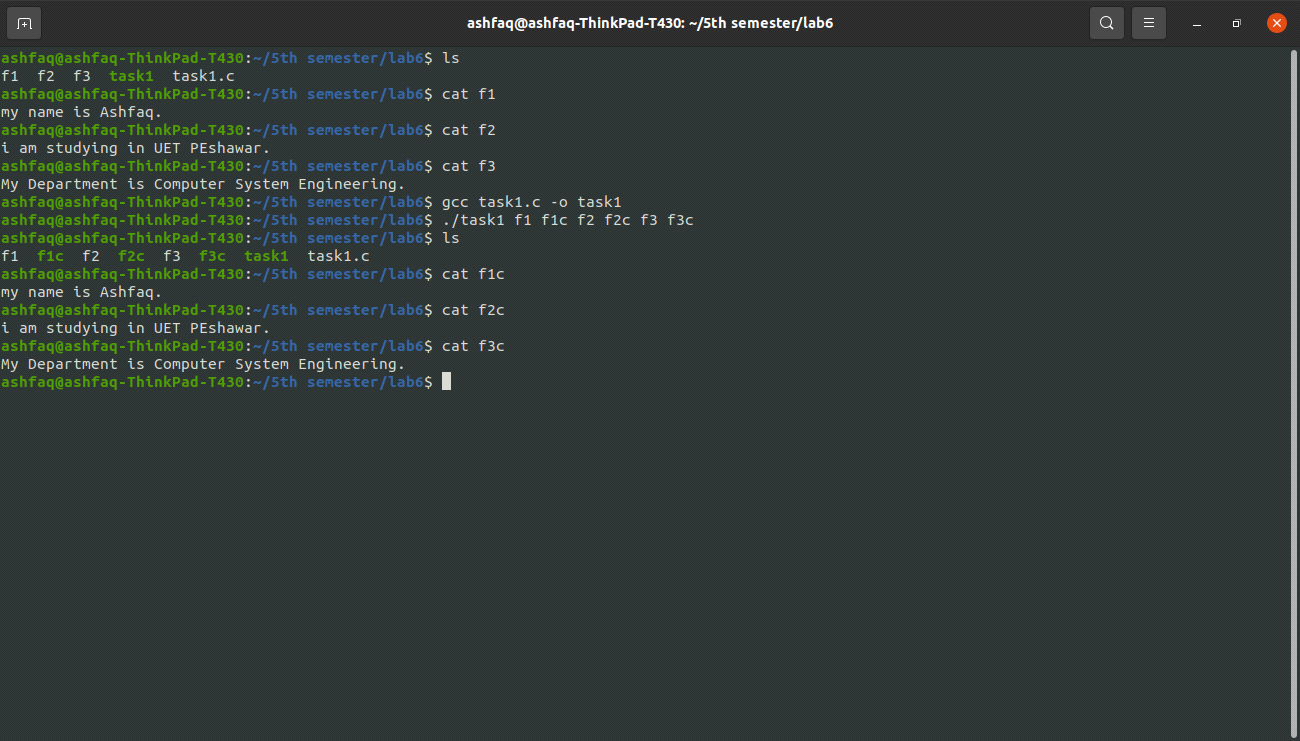
}

}

return 0;

}

Output:



**Task # 02:** Implement “Cat” utility.

1. **Cat** 
   1. Src: STDIN\_FILENO
   2. Dest: STDOUT\_FILENO
2. **Cat file1.txt**
   1. Src: file1.txt
   2. Dest: STDOUT\_FILENO
3. **Cat f1.txt > f2.txt** 
   1. Src: f1.txt
   2. Dest: f2.txt

**Source Code:**

#include <stdio.h>

#include <unistd.h>

#include <stdlib.h>

#include <fcntl.h>

#include <sys/stat.h>

int readwrite(int fd1,int fd2)

{

char buffer[100];

int bytesread;

do

{

bytesread=read(fd1,buffer,sizeof(buffer));

if(bytesread==-1)

{

printf("Sorry! data can't read from %d\n",fd1);

perror("Reason");

}

int byteswritten=write(fd2,buffer,bytesread);

if(byteswritten==-1)

{

printf("Sorry! data can't written on %d\n",fd2);

perror("Reason");

}

}while(bytesread!=0);

};

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

{

if(argc>4 || (argc>1 && argc%2!=0)) //max arguments=4[one this file itself and three from user like "file1 > file1".

// if argc>1 then input argument must be 3 + this file itslef.

{

printf("Sorry! Invalid no of arguments:\n");

return -1;

}

if(argc==1) //mean no user argument then cat will read from 0=STDIN\_FILENO and write on 1=STDOUT\_FILENO.

{

readwrite(0,1);

}

else if(argc==2) //mean one user argument then cat will read from user file and write on 1=STDOUT\_FILENO.

{

int f1=open(argv[1],O\_RDONLY);

if(f1==-1)

{

printf("Sorry! %s can't opened successfully\n",argv[1]);

perror("Reason");

return -1;

}

readwrite(f1,1);

int fd1=close(f1);

if(fd1==-1)

{

printf("Sorry! file %s can't closed successfully\n",argv[1]);

perror("Reason");

}

}

else //mean two argument from user then cat will read from file1 and write on file2.

{

if(\*argv[2]=='>')

{

int f1=open(argv[1],O\_RDONLY);

if(f1==-1)

{

printf("Sorry! %s can't closed successfully\n",argv[1]);

perror("Reason");

return -1;

}

Int f2=open(argv[3],O\_WRONLY|O\_CREAT|O\_TRUNC,S\_IRWXU,S\_IRWXG,S\_IRWXO);

if(f2==-1)

{

printf("Sorry! %s can't closed successfully\n",argv[3]);

perror("Reason");

return -1;

}

readwrite(f1,f2);

int fd1=close(f1);

if(fd1==-1)

{

printf("Sorry! file %s can't closed successfully\n",argv[1]);

perror("Reason");

}

int fd2=close(f2);

if(fd2==-1)

{

printf("Sorry! file %s can't closed successfully\n",argv[3]);

perror("Reason");

}

}

else

{

printf("The Operator %s is Invalid:\n",argv[2]);

return -1;

}

}

return 0;

}

**Output:**

