UNIX I/O

LAB # 06

Fall 2020

CSE302L System Programming Lab

Submitted by: **Shah Raza**

Registration No. : **18PWCSE1658**

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:

Engr. Madiha Sher

Saturday, January 2nd, 2021

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

Lab Objective(s):

- Understand and implement read, write, open, close and unlink function calls.

Task # 01:

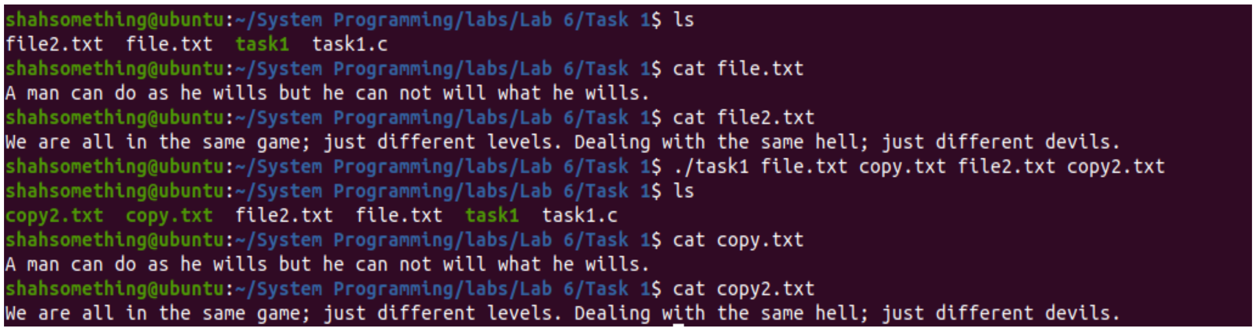
Write a program for parallel file copying using multiple processes.

Code:

```
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/stat.h>
#include <sys/wait.h>


int main(int argc, char *argv[])
{
	if(argc<3 || argc%2==0)
	{
		printf("Invalid number of Arguments\n");
		return -1;
	}
	int x;
	for(int i=1;i<argc;i+=2)
	{
		x= fork();
		if(x==0)
		{
			int fd1 = open(argv[i],O_RDONLY);
			if(fd1==-1)
			{
				perror("Failed to open Source file");
				return -1;
			}

			int fd2 = open(argv[i+1],O_WRONLY | O_CREAT, S_IRWXU | S_IRWXG | 
S_IRWXO);
			if(fd2==-1)
			{
				perror("Failed to open destination file");
				return -1;
			}

			int bytesread;
			char buffer[100];
			do
			{
				bytesread = read(fd1,buffer,100);
				if(bytesread==-1)
				{
```

```
                        perror("Error Occured while reading");
                        return -1;
                    }
                    int byteswriten = write(fd2,buffer,bytesread);
                    if(byteswriten==-1)
                    {
                        perror("Error Occured while writing");
                        return -1;
                    }
                }while(bytesread!=0);

                int cfd1 = close(fd1);
                if(cfd1==-1)
                {
                    perror("Failed to close Source file");
                    return -1;
                }
                int cfd2 = close(fd2);
                if(cfd2==-1)
                {
                    perror("Failed to close destination file");
                    return -1;
                }
                break;
            }
        }
        if(x>0)
        {
            for(int i=1;i<argc/2;i++)
                wait(NULL);
        }

        return 0;
}
```

Output/Results:



```
shahsomething@ubuntu:~/System Programming/labs/Lab 6/Task 1$ ls
file2.txt  file.txt  task1  task1.c
shahsomething@ubuntu:~/System Programming/labs/Lab 6/Task 1$ cat file.txt
A man can do as he wills but he can not will what he wills.
shahsomething@ubuntu:~/System Programming/labs/Lab 6/Task 1$ cat file2.txt
We are all in the same game; just different levels. Dealing with the same hell; just different devils.
shahsomething@ubuntu:~/System Programming/labs/Lab 6/Task 1$ ./task1 file.txt copy.txt file2.txt copy2.txt
shahsomething@ubuntu:~/System Programming/labs/Lab 6/Task 1$ ls
copy2.txt  copy.txt  file2.txt  file.txt  task1  task1.c
shahsomething@ubuntu:~/System Programming/labs/Lab 6/Task 1$ cat copy.txt
A man can do as he wills but he can not will what he wills.
shahsomething@ubuntu:~/System Programming/labs/Lab 6/Task 1$ cat copy2.txt
We are all in the same game; just different levels. Dealing with the same hell; just different devils.
```

Task # 02:

Implement “Cat” utility.

1. Cat

- **a) Src: STDIN_FILENO**
- **b) Dest: STDOUT_FILENO**

2. Cat file1.txt

- **a) Src: file1.txt**
- **b) Dest: STDOUT_FILENO**

3. Cat f1.txt > f2.txt

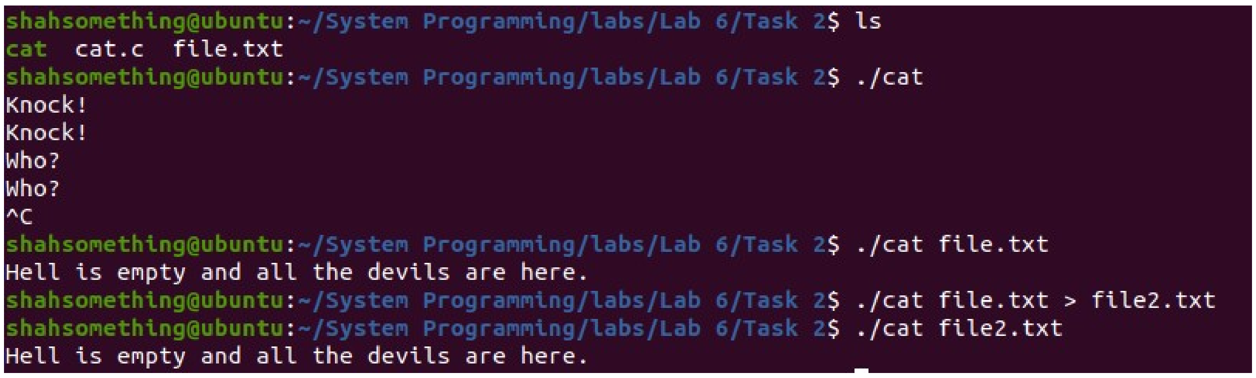
- **a) Src: f1.txt**
- **b) Dest: f2.txt**

Code:

```
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/stat.h>

void ReadWrite(int fd1, int fd2)
{
	int bytesread;
	char buffer[100];
	do
	{
		bytesread = read(fd1,buffer,100);
		int byteswriten = write(fd2,buffer,bytesread);
	}while(bytesread!=0);
}

int main(int argc, char *argv[])
{
	if(argc>4 || (argc>1 && argc%2!=0))
	{
		printf("Invalid number of Arguments\n");
		return -1;
	}

	if(argc==1)
	{
		ReadWrite(0,1);
	}
	else if(argc==2)
	{
		int fd1 = open(argv[1],O_RDONLY);
		if(fd1==-1)
		{
			perror("Failed to open Source file");
			return -1;
		}

		ReadWrite(fd1,1);

		int cfd1 = close(fd1);
		if(cfd1==-1)
		{
			perror("Failed to close Source file");
			return -1;
		}
	}
	else
	{
		if(*argv[2]=='>')
		{
			int fd1 = open(argv[1],O_RDONLY);
			if(fd1==-1)
			{
				perror("Failed to open Source file");
				return -1;
			}

			int fd2 = open(argv[3],O_WRONLY | O_CREAT, S_IRWXU | 
S_IRWXG | S_IRWXO);
			if(fd2==-1)
			{
				perror("Failed to open destination file");
				return -1;
			}

			ReadWrite(fd1,fd2);

			int cfd1 = close(fd1);
			if(cfd1==-1)
			{
				perror("Failed to close Source file");
				return -1;
```

```
                }
            }
            else
                printf("Unabale to recognize %s\n",argv[2]);

        }
        return 0;

}
```

Output:



```
shahsomething@ubuntu:~/System Programming/labs/Lab 6/Task 2$ ls
cat  cat.c  file.txt
shahsomething@ubuntu:~/System Programming/labs/Lab 6/Task 2$ ./cat
Knock!
Knock!
Who?
Who?
^C
shahsomething@ubuntu:~/System Programming/labs/Lab 6/Task 2$ ./cat file.txt
Hell is empty and all the devils are here.
shahsomething@ubuntu:~/System Programming/labs/Lab 6/Task 2$ ./cat file.txt > file2.txt
shahsomething@ubuntu:~/System Programming/labs/Lab 6/Task 2$ ./cat file2.txt
Hell is empty and all the devils are here.
```