**Lab 07 OS Sec E Spring 2018**

**Named Pipes:**

* It is an extension to the traditional pipe concept on Unix. A traditional pipe is “unnamed” and lasts only as long as the process.
* A named pipe, however, can last as long as the system is up, beyond the life of the process. It can be deleted if no longer used.
* Usually a named pipe appears as a file, and generally processes attach to it for inter-process communication. A FIFO file is a special kind of file on the local storage which allows two or more processes to communicate with each other by reading/writing to/from this file.
* A FIFO special file is entered into the filesystem by calling mkfifo() in C. Once we have created a FIFO special file in this way, any process can open it for reading or writing, in the same way as an ordinary file. However, it has to be open at both ends simultaneously before you can proceed to do any input or output operations on it.
* Reading from or writing to a named pipe occurs just like traditional file reading and writing; except that the data for named pipe is never written to or read from a file in hard disk but memory.

**Input/output Redirection:**

**To understand input/output redirection, we need to understand named pipes and dup2 system call first.**

**Dup2:**

Dup2 system call is used to make alias of a file descriptor, for example:

Int fd=open(“file.txt”, O\_WRONLY);

dup2(fd, 1);

cout << “This data will not be printed to the screen, but to the file”, 50);

The above system call will first close file descriptor 1 using close system call. Then, file descriptor fd’s data will be copied to file descriptor 1. So, every write that is made using fd=1 will then be written to file.txt

Similarly,

Int fd=open(“file.txt”, O\_RDONLY);

dup2(fd, 0);

char buffer[10];

cin>>buffer;

Cin will not get input from keyboard, but from file.txt in above piece of code**. (Sample code is given)**

**On shell, we make the input of one command as the output of another command by using | symbol.**

**ls | sort**

ls displays directory contents and sort simply sorts the input data. In above command, we are passing the output of ls command as input to sort command. The output shown on the screen will be generated by sort command. Behind the scenes, it is done by named pipes and dup2.

To redirect the output of a command to a file, we can use > symbol, such as

Ls > 1.txt

The above command will redirect the output generated by ls command to file 1.txt

**Inlab Questions**

**Question 1:**

Create 2 independent programs that perform communication using named pipes. One program will be the client program that will wait for server to send some data via a named pipe. The data sent by the server is as follows:

Operator operand1 operand2

The operands can be +, -, \*, /. The server will then apply the operator on the operands and return the result to client via named pipe. The client will then print the result on the screen

For example, if the server passed the following to the client: + 4 10, then the client will calculate 4+10 and return 14 to server via the pipe. The server will then print it.

**Question 2:**

Implement a program that executes the command:

ls | sort

This will help: execlp("sort","sort",NULL);

**Question 3:**

Write a multithreaded program that calculates various statistical values for a list of numbers. This program will be passed a series of numbers on the command line and will then create three separate worker threads. One thread will determine the average of the numbers, the second will determine the maximum value, and the third will determine the minimum value. For example, suppose your program is passed the integers. (The array of numbers must be passed as parameter to threads, and the thread must return the calculated value to main thread).

90 81 78 95 79 72 85

The main thread will print:

The average value is 82

The minimum value is 72

The maximum value is 95