CST352 S2016

Lab 1

**Purpose:**

In this lab, you will get familiar with writing C code in a Linux environment. You will also explore the Linux process creation and control API.

**Goal:**

Understand and be able to use the Linux process creation and control API.

**Introductory note:**

Treat this write-up as a specification. If I specify something and you do something different, I can take points off. If you think my specification is unclear, ask, don’t guess.

**Part 1**

Write a program called prefilter that reads lines of text from stdin and for each line, prints to stdout all characters including and following a specified character. The character must be specified as a command line argument. For example, given the command:

prefilter x

and the input:

this is x a test

1234x5678

abcde

The program should output the following. Note, the program should NOT output the quotes, I just included those so that whitespace would be obvious.

"x a test"

"x5678"

Note that the third line does not contain an “x”, so the line was removed from the output.

In addition to the output mentioned above, the program should output to stderr the following for each line:

<proc id> pre <char>: <input line>

<proc id> rest: <resulting line>

where <proc id> is the process id of the process (can be obtained using getpid())

<char> is the character being filtered on

<input line> is the actual line read from stdin

<resulting line> is the line including and after <char>

Note: stdout will NOT include blank lines. If the <resulting line> is empty, the “rest:” line should still be written to stderr.

If the program is run without a command line argument, it should print an error to stderr and then terminate. If the command line argument has multiple characters (instead of just one), ignore all but the first.

Hint: Look at the function strchr(). It should prove useful for filtering.

**Part 2:**

Write a program called postfilter that behaves similar to prefilter except that it filters out characters **after** the specified character instead of characters **before** the specified character. For example, the command:

postfilter x

and the input:

this is x a test

1234x5678

abcde

should output the following. Note, the program should NOT output the quotes, I just included those so that whitespace would be obvious.

"this is x"

"1234x"

"abcde"

postfilter should have similar stderr output except that “pre” is replaced by “post”.

Hint: Look at strrchr(). It should prove useful.

**Part 3:**

Write a program called process that will use prefilter and postfilter to process the output of a specified command. Your program must take a variable number of arguments. The first argument specifies the character for the prefilter. The second argument specifies the character for the postfilter. The remaining arguments specify the command whose output is to be filtered. For example, the command:

process x y ls –l

should have the effect of typing the following as a command:

ls -l | prefilter x | postfilter y

**Additional requirements:**

1. You must supply a Makefile to build your executables. The Makefile must have targets for all and clean with the normal meaning.
2. The processes should all be running at the same time. It is not acceptable for the command to finish, and then the prefilter runs and then the postfilter runs. The output should make it clear that all three are running at the same time.
3. Your programs must include paranoid error checking, handling, and (where possible) recovery.
4. Your output must match what is specified in this write-up including white space.
5. You must not have any memory leaks. You must clean up all open files, and any other resources you use.
6. You must wait() for child processes to complete so that you don’t wind up with zombies.

**Other notes:**

1. You will need to use fork(), exec(), and wait() (or their cousins) to create and manage the processes.
2. You will need to use pipe() and dup2() to connect the processes together.

**Submitting your lab:**

To submit your lab, run make clean to clean up your directory, then create a tar file using a command similar to the following:

tar -cvf lab1\_phil\_howard.tar \*

Attach the resulting tar file to BOTH the lab and the lab paranoia Blackboard assignments.