COP 4600 - Homework 1

Due February 15, 2013

Write a C or C++ program which replaces the command shell in Unix.

-After started, it prints a prints a prompt “#” and reads a command line terminated by newline.

-Interpret the following command lines:

# quit

terminates the program

# run command ...

Interprets the first word as the full path to the program to execute, and the others as parameters. It uses fork() + exec() to start the program with the corresponding parameters, and waits until the program terminates (use the waitpid() call).

For instance

 run /usr/bin/xterm

would bring up a terming, but the prompt would not return until the terminal is closed.

Display an error message if the specified program cannot be found or cannot be executed.

# background command...

It is similar to the run command, but it immediately prints the PID of the program it started, and returns the prompt.

# murder PID

Immediately terminate the program with the specific PID (presumably started from this command line interpreter). Use the kill() function call to send a SIGKILL signal to the program. Display success or failure.

To help you, you might want to read a tutorial on

 <http://www.yolinux.com/TUTORIALS/ForkExecProcesses.html>

I have also linked from the webpage a code segment which might help you in reading the commands.

As a note: the easiest way to program this assignment is if you are using a native Unix system, such as Linux. Alternatively, you can use the Cygwin environment in Windows (as far as I know, there is nothing in this project which would prevent you fully implementing it in Cygwin, but your mileage might vary).

## Extra credit (20 points)

For extra credit, implement any of these

* Support for a “for” loop
* Support for a “while” loop
* Support for if-then-else statement.

Note that your implementation does not necessarily follow the same syntax as the one in shells such as bash. Please provide a description and example of use.

## What to submit:

* The code as a single .c or .cpp file.
* If you implemented the extra credit part: a text file describing the syntax of the implementation, and example of use.