

Homework 4

Reading: Man pages; See shell scripting links

Due: Wed, Oct. 4, 2006

Problem 1 (8 Points)

The course Web page has a link to the source code for the test harness for `xsshA`. `xsshA` is a very simple shell language which is a subset of the language `xssh0` which will be implemented in Project A. It is a test harness in the sense that the command sequence is hardcoded into the simple two-dimensional array `cmd[] []` where `cmd[i]` points to the i th command and `cmd[i][j]` points to the j th word of command i .

In the description below, a proper word indicates a metasymbol, square brackets (`[]`) indicate an optional word(s), and `"..."` indicates 0 or more words. `xsshA` supports the following *builtin* (internal) commands:

- `echo [Word] ...`: Display the arguments followed by a newline. Multiple spaces/tabs should be reduced to a single space.
- `quit N`: Quit the shell with an exit status of N .
- `wait N`: The shell should wait for process N to terminate.
- `set Name Value ...`: Set a variable name to a value. A user-defined variable name is a sequence of letters, digits and underscore. There are three special variable names described later: question mark (?), dollar (\$), and exclamation (!). The value of a variable is indicated by preceding the name with the dollar sign. For example, `'set XY 32'` sets the variable `XY` to the string `32`. The value of variable `XY` is denoted by `$XY`. If there is more than one *Value* argument, the values are concatenated together to form a single value; i.e., `'set X 32 ABC'` is equivalent to `'set X 32ABC'`. `'set X ABC $3'` sets the value of `X` to the concatenation of the string `"ABC"` with the value of the variable `3`.

All other commands are assumed to be executables in a directory listed in the `PATH` environment variable.

Here are the other features of `xsshA`:

- a) The command line prompt should be the three character sequence `'>> '` (i.e., `>`, `>`, space).
- b) A non-builtin command is assumed to be a Unix executable that can be found in a directory listed in the `PATH` environment variable.
- c) `$?` is the exit status of the latest process `$$` is the process number of the shell. `#!` is the process number of the last backgrounded process.
- d) All undefined variables have a value of the null string.
- e) The `$` symbol only has a special meaning when it is the first character of a word and it is interpreted as meaning that the value of the variable is desired. So, `XYZ` is the value of the variable `X$YZ` because `$` has no special meaning unless it is the first character of a word.
- f) There are no explicit environment variables available to the user.

- g) An ampersand character (&) at the end of a line indicates that the command should be run in the background.

Note that there is very simple variable substitution, but there is no filename substitution nor command substitution. See `fork(2)`, `waitpid(2)`, `execvp(2)`, `sh(1)`, `gettimeofday(2)`, `exit(3)`. The course Web page also contains links to source code that might be useful to you.

You should fill in the test harness `xsshA.c` so that it can interpret the `xsshA` language. Note that the code recognizes two flags: `-x` and `-d`. The `-x` flag indicates that the command line AFTER variable substitution is displayed. The `-d` flag indicates that debugging output should be displayed on `stderr`. **When debugging is turned on with `-d`, the values returned from each major system call (e.g., `fork`, `wait`, `exec`) should be displayed even if the value is returned in the parameter list (e.g., `waitpid`) and the input parameters to every call to an `exec` function should be displayed.** The debug output should be labeled with the variable names when appropriate so that it is clear what variables are associated with what values. Choose a format that is simple but easy to read.

Submit the following:

- a) Your source code.
- b) The output of the test harness when run with the `-x` and `-d` flags.
- c) For each command, indicate whether your code is working properly or not. If not, indicate what is wrong and what needs to be done to fix the bug(s).