

Homework 3 - Simple Shell

Due by 7:00 p.m. Monday, 3/3/14

For this assignment you will implement your own shell/command-line interpreter (to replace `/bin/bash` and the Windows command prompt). Your shell should read lines of user input into a 4096 byte buffer, then parse and execute the commands by forking off new processes. Write both a POSIX version of your shell that calls `fork()` followed by `execvp()` to create new processes, and a Win32 version that calls `CreateProcess()`. Following each command, your shell should wait for its child process to complete, and then reprint the command prompt. The user should be able to specify the command to execute by giving a path to the executable file (e.g. `./hw1`) or to use path expansion to locate the executable file via the `PATH` environment variable (which is used to construct a series of absolute paths and execute the first file found in this way). Note that the `execvp()` and `CreateProcess()` functions perform this processing automatically; you do not need to program this yourself. Before calling `execvp()` in the POSIX version, your code should parse the input string and separate it into a collection of substrings (stored in `myargv[]`) along with a count of these strings (stored in `myargc`). In the Win32 version, you should be able to pass the input string directly to `CreateProcess()`. If the user enters the `exit` command, your shell should terminate (returning to the regular shell). Here is a sample execution on a Linux machine:

```
Myshell> ls
file1      file2      file3
Myshell> ls -l
total 3
-rwxr-xr-x  1 cassidy      None      2883 Nov  5 12:57 file1
-rwxr-xr-x  1 cassidy      None      1468 Oct 23 14:07 file2
-rwxr-xr-x  1 cassidy      None       200 Jan 24 2013 file3
Myshell> /usr/bin/echo this
this
Myshell> exit
```

You should submit your source code files (one for each platform) and a short writeup in pdf format that includes a description of what you did and the compilation and execution output from each of your programs. Test each version of your program using a command with no arguments, a command with one argument, and a command with two or more arguments. Then use the `exit` command to exit your program and show the output of the same commands in the regular command-line interpreter for that machine. Submit everything to the regular submission link on iLearn, and then submit just the writeup to the TurnItIn link to generate an originality report.