Project 1: Threaded OS Shell - Part B

Objectives

There are four objectives to this assignment:

- To familiarize yourself with the Unix/Linux programming environment.
- To develop your thread programming skills in Java.
- To learn how threads are handled (i.e., starting and waiting for their termination).

Overview

The Shell or Command Line Interpreter is the fundamental User interface to an Operating System. Part b of first project is to enhance the shell you wrote in part a with following added property:

1. Everthing from Part a should work +
2. Each line (typed at the prompt) may contain multiple commands separated with the **;** character. Each of the commands separated by a **;** should be run simultaneously, or concurrently. (Note that this is different behavior than standard Linux shells which run these commands one at a time, in order.) The shell should not print the next prompt or take more input until **all** of these commands have finished executing. For example, the following lines are all valid and have reasonable commands specified:

```
prompt>
prompt> /bin/ls
prompt> ls -l
prompt> ls -l ; cat file
prompt> ls -l ; cat file ; grep foo file2
```

Project1b Requirements

1. Design a simple command line shell that satisfies the above criteria and implement it on venus.
2. Files in the submitted directory could be:
   Project1b.java
   readme

Program Specifications

Your program must be invoked exactly as follows:
java Project1b

Hints

Your shell is basically a loop: it repeatedly prints a prompt parses the input, executes the command specified on that line of input, and waits for the command to finish, if it is in the foreground. You should structure your shell such that it creates a new thread for each new command (except built in commands). There are two advantages of creating a new thread. First, it protects the main shell process from any errors that occur in the new command. Second, it allows easy concurrency; that is, multiple commands can be started and allowed to execute simultaneously (i.e., in parallel style).

Submission : Due: 03/23/2017 at 9:00pm

For this project, you need to hand in two distinct items:

- **Your source code (no object files or executables, please!)**
- **A `readme` file as stated above**
- Under your **`$HOME`** directory, create a subdirectory: **`osproject1b`** **(case sensitive)**. Copy all of your .java source files into this directory. Do **not** submit any other files. **MAKE SURE TO GIVE ME PROPER PERIMISSIONS AS DISCUSSED IN CLASS**. On the deadline as listed above - system will copy all the files from your respective directory to mine for grading **Remember: No late projects will be accepted!**

Criteria (50)

- Submission of required files only, with name, student number etc on all submitted files (5)
- Warning free compilation and linking of executable with proper name (5)
- Threaded External command functionality (25)
- Readability, suitability & maintainability of source code (15)

Ensure that you address all the points specified in this project documentation. Finally, while you can develop your code on any system that you want, make sure that your code runs correctly on venus. Specifically, since libraries and environments sometimes vary in small and large ways across systems, you should verify your code executes on venus. This machine is where your projects will be tested.