CS 464/564, Assignment 1

This assignment can be solved in teams of at most *two* students,
is worth *two tokens*, and
is due on *31 January at 11:59 pm*

This assignment asks you to build simple Unix shell and gives you in the process an opportunity to demonstrate that you are mastering concurrency (including running external executables) and file manipulation using the POSIX API.

As a fair warning, there are many, many shells like this one floating all over the Web, but before being inclined to inspire yourselves from them note that the requirements are substantially different in at least two respects and not observing the *current* requirements will result in poor marks. In other words, make sure you follow the specifications herein, not something else.

Our shell repeatedly provides a prompt, accepts a command (possibly with arguments), and executes it. Most commands are external executables, but some are internal.

The shell ignores completely its environment and reads a simple configuration file upon startup instead. The file is named `shconfig` and resides in the current directory. It contains pairs consisting in a variable name and a value for that variable, separated by at least one blank. The two interesting lines in this file have the form

```
VSIZE v
HSIZE h
```

where v and h are strings representing positive integers. These variables are used by the internal command `more`. If these lines do not exist (or if the file is missing altogether), two such lines must be appended to the configuration file (which must be created first if not already there) with the values 40 for v and 75 for h. The configuration file may contain other lines, which should be ignored.

The following functionality should then be implemented:

- All the typed in commands should be considered external *except* the commands `more` and `exit`. Your shell should attempt to execute these commands. The search path for the external commands contains only the directories `/bin` and `/usr/bin` (so any command outside these directories should be ignored unless it is given by absolute path). If the command cannot be executed, then the shell should print an appropriate message; if the command produces an error during its execution, then the shell should print yet another appropriate message *in addition to whatever is printed by the command itself.*

  The arguments for the commands are provided in the same manner as in normal shells.

  A command may be *prefixed* by the character `&` followed by a blank. If this is the case, then the command is executed in *background*, in the sense that the shell presents the prompt (and accepts another command) before the completion of the previous command. Whenever the background command terminates (with or without errors) the shell *must print an appropriate* *message, again in addition to whatever is printed by the command*. This additional message must be printed at the time of termination (not later like is the case in the usual shells).

- The command `more` must be implemented internally. This command receives one file name as argument, and displays it to the standard output page by page. The number of columns in a page and the number of characters in a column are taken from the configuration file and are given by the value v of variable `VSIZE` and by the value h of variable `HSIZE`, respectively.

  The command `more` displays v lines of the file given as argument and then awaits user input. If the input is an empty line (that is, just a `'\n'`) then the next v lines are displayed; if the input is anything else, then the command terminates. The command also terminates when it reaches the end of the file. All the lines are displayed truncated to h characters if necessary (i.e., if they are longer than h characters).

- The command `exit` terminates the shell.

What to submit For this assignment you *must* use the POSIX interface presented in class for file access rather than library functions.

The default target of your makefile must produces an executable named `sshell` and residing in the root directory of your submission. Please read carefully the submission requirements on the course's Web site; failure to observe all of these requirements may have serious consequences for your marks.