Bash-like Shell

Design

The shell comprises of 4 modules - a parser module, a process module, a job module and the shell module itself. The parser takes in the user input and simulates a finite state machine. It determines if the user input is valid, and the type of job that needs to be spawned - a single process job, or a multi-process job using either a pipe, a message queue or shared memory for communication.

A job is a group of processes communicating through a pipe, a message queue or shared memory. The job module stores information related to a particular command - the number of processes, the communication medium (pipes, message queues or shared memory), whether it is a foreground or background job, status etc. It also stores the filenames to which stdin/stdout are redirected (if they are).

The processes module contains all the information necessary to launch a process using the `execvp` system call. It stores the name, arguments and file descriptors of remapped stdin/stdout when required.

The shell module invokes the functions of the above modules. It performs job control and manages foreground and background jobs.

Shell Builtins

- `exit`
- `clear`
- `jobs`
- `bg`
- `fg`
- `daemonize`

Features

- Can execute any program with arguments found in the `PATH` environment variable using the `execvp` system call.

- Implements full job control. Jobs can be launched in the background with `&`. Foreground jobs can be suspended by sending SIGSTOP (Ctrl + Z). Suspended jobs can be resumed with `fg` or `bg`. A list of running and suspended background jobs can be viewed with `jobs`.

- Implements pipelining of any number of processes using pipes. Also supports redirection of stdin/stdout to file. For example: `ls | sort | wc -l > outfile`.

- Supports the `##` operator - `ls ## wc , sort` indicates the output of `ls` is mapped to the input of both `wc` and `sort` using message queues.

- Supports the `SS` operator - `ls SS wc , sort` indicates the output of `ls` is mapped to the input of both `wc` and `sort` using shared memory.

- Supports the `daemonize` builtin - launches any process as a daemon.

- Supports the standard `<`, `>` and `>>` redirection operators.