User Guide

Áron Pákozdi

December 2024

1 Goal of the program

In the homework assignment I chose to create my own command line based shell on unix based operating systems. Because of the different system calls of Linux and Windows, I cannot create my program to be cross-platform. Its functionality should be similar to the already existing ones. A shell is a program, that serves as an interface between the user and the operating system. It allows the user to interact with the computer by entering commands into a terminal emulator.

2 Instructions

My shell behaves similarly to already existing popular command line based shell programs like bash or zsh.
My shell supports two modes: interactive mode and non-interactive mode.

2.1 Interactive mode

To run the shell in interactive mode type
`prompt$ ./kpsh`
into the terminal.
Then the shell program should be running, indicating it by printing a prompt and waiting for inputs. There are infinitely many possible inputs to write in a shell program, but here are some examples:
Opening the manual:
`prompt$ man man`
This should open the manual of the man command

Printing to the terminal: `prompt$ echo "hello, world!"`
Prints hello, world! to the terminal.

Getting current location (current working directory):
`prompt$ pwd`

Listing all files and directories in current directory:
`prompt$ ls -a`

The program also supports I/O redirection (although not working as intended). There are 7 redirectors:

- '`>`' : Redirects standard output to a file. Creates new file if it does not exist, overwrites if exists
- '`>>`' : Redirects standard output to a file. Creates new file if it does not exits, appends to the end if exists.
- '`<`' : Redirects standard input. Reads input from a file
- '`<<`' : Redirects standard input. Reads input from multiple files
- '`2>`' : Redirects standard error to a file. Creates new file if it does not exist, overwrites if exists
- '`2>>`' : Redirects standard error to a file. Creates new file if it does not exist, appends to the end of the file if exists.
- '`|`' : pipe, redirects a commands output to another commands input.

2.2 Non-interactive mode

To run commands in Non-interactive mode type for example:
`prompt$ "/bin/ls" | ./kpsh`
The output should be the same as running `ls` in interactive mode.

3 File handling

The program uses file handling with the redirections. If the user redirects an output to a file, or reads a file. The file's location is given by the user, and written or created anywhere. Same with reading.