

This screenshot illustrates basic functionality of the shell. Both ls and ps work initially since they can be found in /bin. However, nl needs a full path to be specified in order to work. 

Here I set the path variables for ease later on as well as illustrate the history functionality. Ls is run again since I want to run the first command again.



This screenshot shows both redirections and pipes working simultaneously in action. Only the lines inside messages.txt containing a 1 are printed.



Example screenshot of multiple pipes working.

The way my shell executes (generally) is as follows. After getting the standard input, the shell transforms the input so that there are spaces in between all strings and special characters (<,>,|,&). After transforming the input, the resulting buffer is tokenized in order to count the number of total arguments, special characters, and set redirections if needed for built in functions. Then the shell checks to see if the user input calls any of the built in functions, launching them if needed. After no built-ins match the input, the shell organizes the buffer to count the number of args for a single command or count the total number of commands as well as their arguments if multiple are being run in one line. After counting arguments and setting up parameters, the command is finally sent to a function to be run. The launchFun() function is the function that spawns child processes to run any funcitons that are not built-ins. If there are no special characters a child is spawned and runs the passed program. If there are special characters, launchFun checks to see which special characters are passed and implements the appropriate pipes, redirects, or runs the program in the background. After the parent waits for all functions to finish, control is returned to the user.