Programming Assignment 1

CSE 3320.002
Due: October 10, 2016 11:59PM

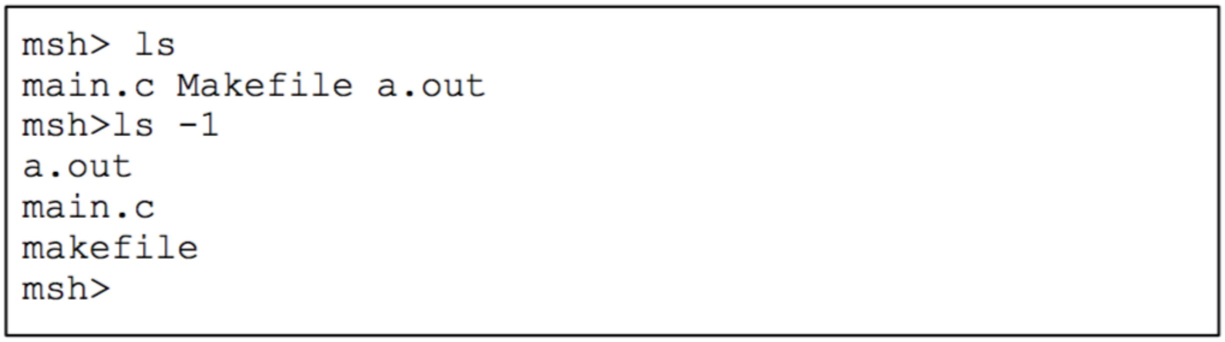
Description

In this assignment you will write your own shell program, Mav shell (msh), similar to bourne shell (bash), c-shell (csh), or korn shell (ksh). It will accept commands, fork a child process and execute those commands. The shell, like csh or bash, will run and accept commands until the user exits the shell.

Your file must be named msh.c

Functional Requirements

Requirement 1: Your program will print out a prompt of msh> when it is ready to accept input. It must read a line of input and, if the command given is a supported shell command, it shall execute the command and display the output of the command.



```
msh> ls
main.c Makefile a.out
msh>ls -1
a.out
main.c
makefile
msh>
```

Requirement 2: If the command is not supported your shell shall print the invalid command followed by ": Command not found."

```
msh> lss
lss: Command not found.
msh>
```

Requirement 3: If the command option is an invalid option then your shell shall print the command followed by ": invalid option --" and the option that was invalid as well as a prompt to try —help.

```
msh> ls -j
ls: invalid option -- 'j'
Try `ls —help' for more information
msh>
```

`exec()` will give you this output make sure you pass it on to your user.

Requirement 4: After each command completes, your program shall print the msh> prompt and accept another line of input.

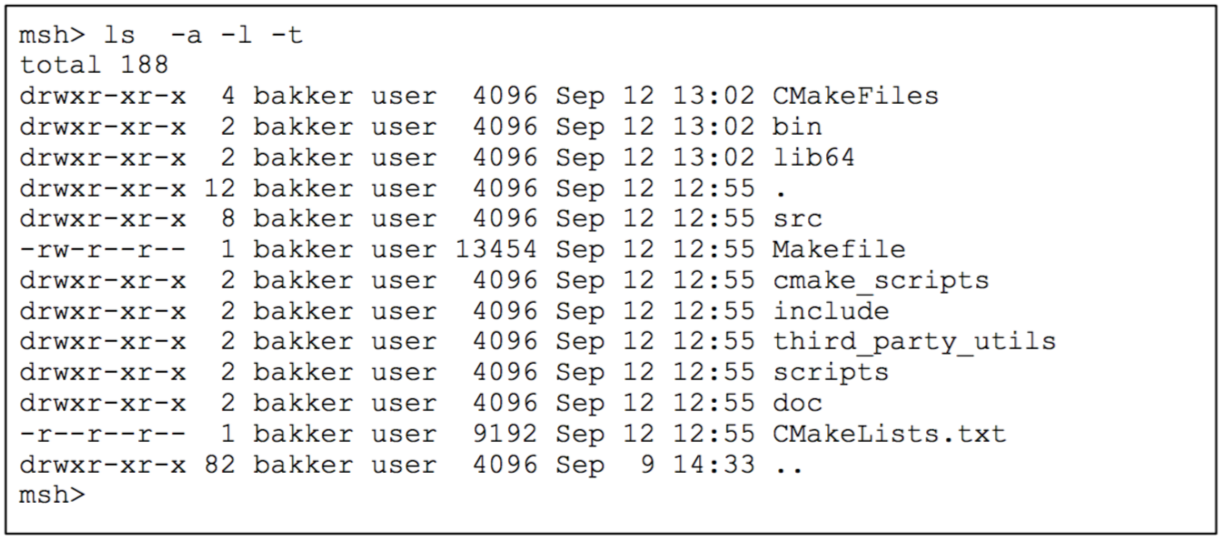
Requirement 5: Your shell will exit with status zero if the command is "quit" or "exit".

Requirement 6: If the user types a blank line, your shell will, quietly and with no other output, print another prompt and accept a new line of input.

```
msh>
msh>
```

Commands and Optional Parameters

Requirement 7: Your version of Mav shell shall support up to five command line parameters in addition to the command.



```
msh> ls  -a -l -t
total 188
drwxr-xr-x  4 bakker user  4096 Sep 12 13:02 CMakeFiles
drwxr-xr-x  2 bakker user  4096 Sep 12 13:02 bin
drwxr-xr-x  2 bakker user  4096 Sep 12 13:02 lib64
drwxr-xr-x 12 bakker user  4096 Sep 12 12:55 .
drwxr-xr-x  8 bakker user  4096 Sep 12 12:55 src
-rw-r--r--  1 bakker user 13454 Sep 12 12:55 Makefile
drwxr-xr-x  2 bakker user  4096 Sep 12 12:55 cmake_scripts
drwxr-xr-x  2 bakker user  4096 Sep 12 12:55 include
drwxr-xr-x  2 bakker user  4096 Sep 12 12:55 third_party_utils
drwxr-xr-x  2 bakker user  4096 Sep 12 12:55 scripts
drwxr-xr-x  2 bakker user  4096 Sep 12 12:55 doc
-r--r--r--  1 bakker user  9192 Sep 12 12:55 CMakeLists.txt
drwxr-xr-x 82 bakker user  4096 Sep  9 14:33 ..
msh>
```

You do not have to support redirection or pipes: “>”, “<”, “|”. You do not have to support backgrounding the process with “&”

Requirement 8: Your shell shall support and execute any command entered. Any command in `/bin`, `/usr/bin/`, `/usr/local/bin/` and the current working directory is to be considered valid for testing.

Your shell shall search in the following PATH order:

1. Current working directory,
2. /usr/local/bin
3. /usr/bin
4. /bin

Parameters may also be combined. For example, ps may be executed as: `ps –aef` or `ps –a –e -f`

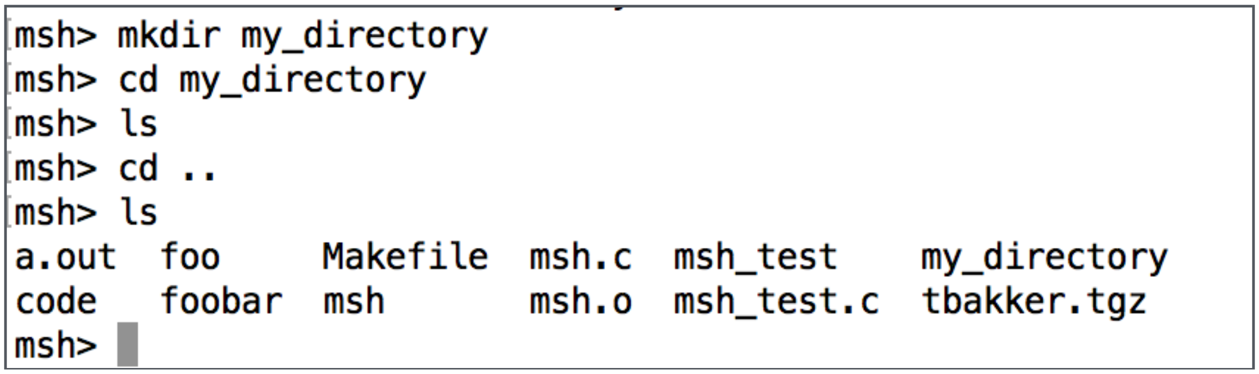
Requirement 9: Mav shell shall be implemented using fork(), wait() and one of the exec family of functions. To find which exec functions you can use on cse3320, type:

```
man 3 exec
```

Your Mav shell shall not use system(). **Use of system() will result in a grade of 0.**

Requirement 10: Typing ctrl-c or ctrl-z shall not stop or kill your shell. Your shell shall ignore those keystrokes.

Requirement 11: Your shell shall support the cd command to change directories



```
msh> mkdir my_directory
msh> cd my_directory
msh> ls
msh> cd ..
msh> ls
a.out  foo     Makefile  msh.c  msh_test    my_directory
code   foobar  msh       msh.o  msh_test.c  tbakker.tgz
msh> 
```

Requirement 12: Your shell shall support the shopped command to list the PIDs of the last 10 processes spawned by your shell

```
msh> showpid
      5216
      5472
      5476
      6123
     7194
     7392
     7812
```

Nonfunctional Requirements

Requirement 13: Your source file shall be named msh.c.

Requirement 14: Your source files must be ASCII text files. No binary submissions will be accepted.

Requirement 15: Tabs or spaces shall be used to indent the code. Your code must use one or the other. All indentation must be consistent.

Requirement 16: No line of code shall exceed 100 characters.

Requirement 17: Each source code file shall have the following header filled out:

```
/*
 * Name:
 * ID #:
 * Programming Assignment 1
 * Description:
 */
```

Requirement 18: All code must be well commented. This means descriptive comments that tell the intent of the code, not just what the code is executing. The following explains the intent:

```
    // We are going to move the working_str as we process
    // the string the pointer points towards so we need to
    // keep track of its original value so we can deallocate
    // the correct amount at the end
    char *working_root = working_str;
```

The following just describes to code and provides no real value and will earn less credit:

```
    // Store the value of the working string pointer
    char *working_root = working_str;
```

When in doubt over comment your code.

Requirement 19: Keep your curly brace placement consistent. If you place curly braces on a new line , always place curly braces on a new end. Don't mix end line brace placement with new line brace placement.

Requirement 20: Each function should have a header that describes its name, any parameters expected, any return values, as well as a description of what the function does. For example :

```
/*
 * Function: processSystemTime
 * Parameter(s): time_delta - A float representing the time drift
 * since the system was last queried. Normalized (0-24 hrs) to
 * 0.0-1.0
 * Returns: An integer value of 0 for success and -1 for error
 * Description: Processes the time value drift calculated from
 * the last received satellite update.
 */
int processSystemTime( float time_delta )
```

Requirement 21: Remove all extraneous debug output before submission. The only output shall be the output of the commands entered or the shell prompt.

Hints and Suggestions

Read the manual pages for the following system and library calls at a minimum:

```
fork, exec, exit, printf, fgets, strtok or
strsep,  strcmp, wait
```

Use fork and one of the exec family as discussed in class to execute the command and call wait to wait for the child to complete. If the command is “cd” then use chdir() instead of exec. Note, chdir() must be called from the parent.

If you see garbage in any of your commands or parameters, try using the functions memset() or bzero() to clear out your input string and token array before and/or after you are done using them. Also, verify you are NULL terminating your strings.

Getting the source

To get the code for assignment 1:

1. Change to your `cse3320-projects` directory

2. Enter the command:

   ```
   git pull /usr/local/share/CSE3320/cse3320-projects/
   ```

   Do not enter `git clone`.

3. The source for the assignment will be in the `Assignment1` directory.

Building the source

1. Change to your `cse3320-projects/Assignment1` directory.

2. Type: `make`

Running the executable

1. Type: `./msh` and press <return>

Grading

The assignment will be graded out of 100 points. Code that does not compile will earn 0.

How to submit homework

1. From your cse3320-projects/assignment1/ directory type:

```
make submit assignment=1
```

Grading rubric

Requirement	Points Possible
Command parsing	15
Executes any command	20
No zombies (waitpid)	5
cd command supported	10
showpid command supported	10
Forks correctly	15
ctrl-c and ctrl-z caught	5
Exit correctly	5
Code formatted correctly	5
No extra/debug output	5
Program submitted correctly	5
Total Points	**100**

Academic Integrity

This assignment must be 100% your own work. No code may be copied from friends, previous students, books, web pages, etc. All code submitted is checked against a database of previous semester's graded assignments, current student's code and common web sources. Code that is copied from an external source will result in a 0 for the assignment and referral to the Office of Student Conduct.