

**ELEC 377 Lab 2 – Testing Plan**

Section 003, Group 16 – Thursday Lab

Erhowvosere Otubu, #20293052

Ivan Samardzic, #20296563

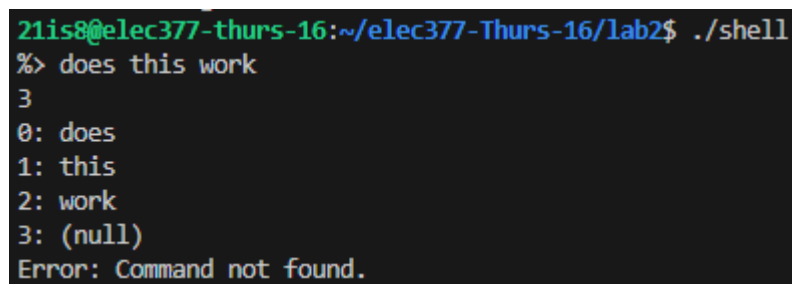
Date of Submission: October 17, 2023

## Testing

The following sections describe multiple test cases used to confirm the validity and accuracy of the sample shell. Testing was conducted in accordance with the Lab 2 outline and is split up into separate testing methods, as shown below.

### Case 1 – Split Command Line

To test the accuracy of splitting up inputted arguments from the command line, the debugging code was uncommented. An arbitrary set of arguments was used for this purpose. This was initially run in the shell, and printed to the command line. This output is shown in Figure 1, below.

A terminal window with a black background and green text. The prompt is '21is8@elec377-thurs-16:~/elec377-Thurs-16/lab2\$'. The user enters './shell'. The prompt changes to '%>'. The user enters 'does this work'. The output shows the number of words '3', followed by the index and word for each token: '0: does', '1: this', '2: work', and '3: (null)'. Finally, it prints 'Error: Command not found.'.

```
21is8@elec377-thurs-16:~/elec377-Thurs-16/lab2$ ./shell
%> does this work
3
0: does
1: this
2: work
3: (null)
Error: Command not found.
```

*Figure 1: Demonstration of splitting the command line within the terminal.*

Visibly, the program successfully printed the number of words, their respective index in the array, and the null terminator. The splitting up of the command line is also proven to be accurate given all other output files, where the commands are accurately divided and executed. This is deemed successful as a result.

### Case 2 – Change Directory

To test the accuracy of the change directory (cd) command, "cd" was inputted into the terminal. This was initially run in the shell, and printed to the command line. Upon execution, the pwd command was also run to display the accurate update in working directory after completion of cd. This output is shown in Figure 2, below.

```
21is8@elec377-thurs-16:~/elec377-Thurs-16/lab2$ ./shell
%> cd
%> pwd
/home/21is8
%> cd elec377-Thurs-16
%> pwd
/home/21is8/elec377-Thurs-16
%> cd lab77
Error: Unsuccessful.
%> exit
```

*Figure 2: Demonstration of cd within the terminal.*

Upon the initial cd, the directory was changed to the home directory “/home/21is8”. Next, cd was run with an additional argument for a specific path, which was “elec377-Thurs-16”. Inputting pwd now shows the updated current directory “/home/21is8/ elec377-Thurs-16”. Finally, an error case was run to display the accurate handling of invalid directory name. This is deemed successful as a result.

Following this test, the script was executed in an external output file, “testCD.txt”, as seen in the repository. This execution was deemed successful, since the text file displays the exact commands used above. This includes the single argument cd, two argument cd, and error case for invalid path name. The pwd was run in between each cd to display the update in directories. The shell was then terminated with the “exit” command.

### Case 3 – List Files

To test the accuracy of the list files (ls) command, “ls” was inputted into the terminal. This was initially run in the shell, and printed to the command line. This output is shown in Figure 3, below.

```
21is8@elec377-thurs-16:~/elec377-Thurs-16/lab2$ ./shell
%> ls
hello.c
Makefile
shell.c
shell
testHelloFile.txt
hello
%> ls -a
.
hello.c
..
Makefile
shell.c
shell
testHelloFile.txt
hello
%> ls -b
Error: Incorrect usage of ls command.
```

Figure 3: Demonstration of ls within the terminal.

Upon the initial ls, all files within the “/elec377-Thurs-16/lab2” directory were accurately listed. Next, ls was run with an additional argument “-a” to display all hidden files (beginning with ‘.’). Finally, an error case was run to display the accurate handling of invalid ls command. This is deemed successful as a result.

Following this test, the script was executed in an external output file, “testLS.txt”, as seen in the repository. This execution was deemed successful, since the text file displays the exact commands used above. This includes the single argument ls, two argument ls -a, and error case for invalid ls command. The shell was then terminated with the “exit” command.

#### Case 4 – Print Working Directory

To test the accuracy of the print working directory (pwd) command, “pwd” was inputted into the terminal. This was initially run in the shell, and printed to the command line. This output is shown in Figure 4, below.

```
21is8@elec377-thurs-16:~/elec377-Thurs-16/lab2$ ./shell
%> pwd
/home/21is8/elec377-Thurs-16/lab2
%> pwd fff
Error: Too many args for pwd command.
```

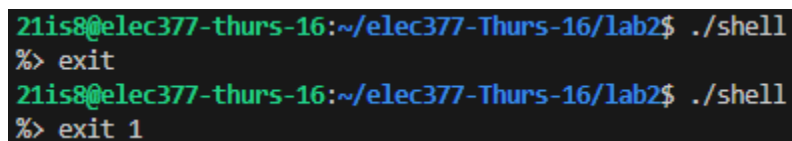
Figure 4: Demonstration of pwd within the terminal.

Upon the initial `pwd`, the working path `“/elec377-Thurs-16/lab2”` was accurately printed. Finally, an error case was run to display the accurate handling of invalid `pwd` with too many arguments. This is deemed successful as a result.

Following this test, the script was executed in an external output file, `“testPWD.txt”`, as seen in the repository. This execution was deemed successful, since the text file displays the exact commands used above. This includes the single argument `pwd`, and error case for invalid `pwd` command. The shell was then terminated with the `“exit”` command.

### Case 5 – Exit Program

To test the accuracy of the termination (`exit`) command, `“exit”` was inputted into the terminal. This was initially run in the shell, and printed to the command line. This output is shown in Figure 5, below.



```
21is8@elec377-thurs-16:~/elec377-Thurs-16/lab2$ ./shell
%> exit
21is8@elec377-thurs-16:~/elec377-Thurs-16/lab2$ ./shell
%> exit 1
```

*Figure 5: Demonstration of exit within the terminal.*

Upon the initial single argument `exit` command, the program terminated as expected. Next, `exit` was inputted with an additional argument (code 1) to display the accurate handling of a 2-argument `exit` command. The program terminated with exit code 1, and the command is now deemed successful as a result.

Following this test, the script was executed in an external output file, `“testEXIT.txt”`, as seen in the repository. This execution was deemed successful, since the text file displays accurate termination. Another file script was executed into `“testEXIT1.txt”`, in which an error case of more than two arguments was handled, and then an `exit` with code 1 was performed after.

### Case 6 – File From Current Working Directory

To test the accuracy of running an external file from the current working directory, the command `“./filename”` was used. Specifically, the file `“hello.txt”` was used as input to the command. This was initially run in the shell, and printed to the command line. This output is shown in Figure 6, below.

```
21is8@elec377-thurs-16:~/elec377-Thurs-16/lab2$ make shell
cc -o shell -g shell.c
21is8@elec377-thurs-16:~/elec377-Thurs-16/lab2$ ./shell
%> ./hello
Hello 21is8
```

Figure 6: Demonstration of running an external file within the terminal.

Visibly, the call to “hello.txt” successfully printed the contents of the file (i.e “hello”, user id).

Following this test, the script was executed in an external output file, “testHelloFile.txt”, as seen in the repository. This execution was deemed successful, since the text file displays the command “./hello”, as done previously in the command prompt, and outputs the exact contents of the “hello.txt” file. To test an error case, the the command “./hello1” was then executed to attempt to open a non-existent file “hello1” in the working directory. The program could not open this file, as expected, and displayed a subsequent error message. The shell was then terminated with the “exit” command.

### Case 7 – Standard Unix Commands

To test the accuracy of the standard unix commands, a change directory was preformed into the path “/usr/bin”. In this directory, the user is given access to all the standard unix commands. To test this, the command “cat /etc/passwd”, as used in lab 1, was executed. This was initially run in the shell, and printed to the command line. This output in shown in Figure 7, below.

Visibly, the cd to “/usr/bin” allowed for the execution of any commands from this list. The command “cat /etc/passwd” was simply used to prove that the user has access to a whole new set of commands.

Following this test, the script was executed in an external output file, “testUSRBIN.txt”, as seen in the repository. This execution was deemed successful, since the text file displays accurate cd into the unix commands directory, and the subsequent execution of an arbitrary command such as “cat /etc/passwd”. The shell was then terminated with the “exit” command.

```

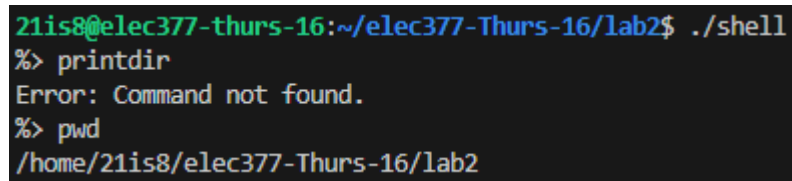
21is8@elec377-thurs-16:~/elec377-Thurs-16/lab2$ ./shell
%> cd /usr/bin
%> cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
systemd-timesync:x:102:104:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:106:./nonexistent:/usr/sbin/nologin
syslog:x:104:110:./home/syslog:/usr/sbin/nologin
_apt:x:105:65534:./nonexistent:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uidd:x:107:112:./run/uidd:/usr/sbin/nologin
tcpdump:x:108:113:./nonexistent:/usr/sbin/nologin
sshd:x:109:65534:./run/sshd:/usr/sbin/nologin
landscape:x:110:115:./var/lib/landscape:/usr/sbin/nologin
pollinate:x:111:1:./var/cache/pollinate:/bin/false
fwupd-refresh:x:112:116:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
systemd-coredump:x:999:999:systemd Core Dumper:./usr/sbin/nologin
sysops:x:1000:1000:./home/sysops:/bin/bash
lxd:x:998:100:./var/snap/lxd/common/lxd:/bin/false
ajg17:x:1001:1000:./home/ajg17:/bin/bash
jas21:x:1002:1000:./home/jas21:/bin/bash
trd:x:1003:1000:./home/trd:/bin/bash
wht:x:1004:1000:./home/wht:/bin/bash
21eo4:x:1007:1000:./home/21eo4:/bin/bash
21is8:x:1008:1000:./home/21is8:/bin/bash

```

Figure 7: Demonstration of standard unix commands within the terminal.

## Case 8 – Commands Not Recognized

To test the accuracy of the error cases within the developed script, a command that does not exist (printdir) was attempted. This was initially run in the shell, and printed to the command line. This output is shown in Figure 8, below.



```
21is8@elec377-thurs-16:~/elec377-Thurs-16/lab2$ ./shell
%> printdir
Error: Command not found.
%> pwd
/home/21is8/elec377-Thurs-16/lab2
```

*Figure 8: Demonstration of unrecognized commands within the terminal.*

Visibly, the invalid command was met with an appropriate error message. A valid command such as “pwd” was then executed directly after this to prove that the program can differentiate between invalid and valid commands.

Following this test, the script was executed in an external output file, “testUNRECOGNIZED.txt”, as seen in the repository. This execution was deemed successful, since the text file displays identical error handling as exhibited within the terminal. It then also displays that a valid command can be executed directly after. The shell was then terminated with the “exit” command.