ELEC 377- Operating Systems

Lab 2: Write a Simple Shell

Testing Document

Bradley Stephen & Ben Wyand

Introduction

To make sure the shell program worked as expected, we tested both internal and external commands. The testing was done using the script command, which captured all input and output. Our goal was to confirm proper command parsing, the execution of internal commands, handling of external programs, and error management.

Testing the splitCommandLine() Function

The splitCommandLine() function breaks down user input into individual arguments. We tested it using the input "this is a sentence" to check if it correctly splits the input into tokens.

Test Command:

```
%> this is a sentence
```

Expected Output:

```
4 arguments found
0: this
1: is
2: a
3: sentence
4: (null)
```

Explanation:

The function successfully identified 4 arguments. Each word was stored in the args[] array, and the last element was NULL, which is essential for functions like execv() that need a null-terminated argument array.

Testing Internal Commands

1. **Testing pwd:**

- **Command:** pwd
- **Expected Output:** The current working directory should be printed.

Recorded Output:

```
%> pwd
/home/19bbs2/elec377-tues-1/lab2
```

Explanation:
The output confirms that pwd successfully printed the working directory, verifying that the command works correctly.

2. **Testing cd:**
    - **Command:** cd elec377-tues-1
    - **Expected Output:** Should change the directory to elec377-tues-1.

Recorded Output:

```
%> cd elec377-tues-1
%> pwd
/home/19bbs2/elec377-tues-1
```

Explanation:
cd correctly changed to the elec377-tues-1 directory and pwd confirmed the current working directory was updated. No errors were encountered.

3. **Testing cd ..:**
    - **Command:** cd ..
    - **Expected Output:** Moves up one directory level.

Recorded Output:

```
%> cd ..
%> pwd
/home/19bbs2
```

Explanation:
cd .. worked as expected, moving up one directory, with pwd confirming the new directory.

4. **Testing cd to a non-existent directory:**
    - **Command:** cd elec377-tues-1/lab
    - **Expected Output:** Should print an error message since the directory does not exist.

Recorded Output:

```
%> cd elec377-tues-1/lab
chdir: No such file or directory
```

Explanation:
The shell correctly handled the error, printing a message indicating the directory doesn't exist.

5. **Testing ls:**
   - **Command:** ls
   - **Expected Output:** Lists files and directories in the current working directory.

   Recorded Output:

   ```
   %> ls
   Makefile
   hello
   hello.c
   shell
   shell.c
   test1.txt
   ```

   Explanation:
   The ls command correctly listed all files and directories in the current directory. It proves that the shell handles internal commands like ls as expected.

6. **Testing exit:**
   - **Command:** exit
   - **Expected Output:** Exits the shell and terminates the program.

   Recorded Output:

   ```
   %> exit
   ```

   Explanation:
   The exit command worked as intended, terminating the shell process.

Testing External Commands

1. **Executing a Local Program (./hello):**
   - **Command:** ./hello
   - **Expected Output:** The program should execute and Greet user by name.

Explanation:
The shell successfully executed an external program located in the current directory and printed “Hello 19bbs2”.

2. **Running a System Command (/usr/bin/echo):**
    - **Command:** /usr/bin/echo Hello from the shell!
    - **Expected Output:** Displays "Hello from the shell!".

    Explanation:
    The shell executed an external command from /usr/bin and printed the expected output.

3. **Testing an Invalid Command:**
    - **Command:** notarealcommand
    - **Expected Output:** Displays an error message indicating that the command wasn't found.