Lab 4: Shell Scripting

Nathan Jarus

June 12, 2017

Introduction

This lab will give you some experience writing shell scripts. You will need to sign in to https://git.mst.edu and `git clone` the repository for this lab. Your repository will be named something along the lines of `2017SS-A-lab04-nmjxv3`. Make sure to clone with the HTTPS URL (unless youve set up SSH keys).

I strongly advise you to experiment and to test your code as you go along! Bash is a little weird, so checking to make sure it's doing what you think it is is important.

Problem 1: Menus and Files

Make a shell script named `menu.sh` that loops through all the files in the current directory and for each file, prints out a menu:

```
v) View file
e) Edit file
c) Compile file
q) Quit
```

Then the script gets the user's choice and does it. Consult the following list to see what 'it' is:

- View file: Open the file with `less`
- Edit file: Open the file in a text editor (your choice which one)
- Compile file: Compile the file with `g++`
- Quit: Bail out of the loop with `break` or `exit`
- Anything else: Print an error message and go to the next file

For example, if the current directory contains `file.cpp`, `file.h`, `goog.sh`, and `menu.sh`, your output might look something like this:

```
$ bash menu.sh
v) View file.cpp
e) Edit file.cpp
c) Compile file.cpp
q) Quit
e

v) View file.h
e) Edit file.h
c) Compile file.h
q) Quit
v

v) View goog.sh
e) Edit goog.sh
c) Compile goog.sh
q) Quit
f
INVALID RESPONSE

Skipping this file!

v) View menu.sh
e) Edit menu.sh
c) Compile menu.sh
q) Quit
q
```

Hints and requirements:

- Your script needs at least one function
- You should probably use a case statement!
- `for file in *.txt` loops over all `.txt` files in the current directory.

Problem 2: Your Own (Terrible) Search Engine

Write a bash script named `goog.sh` that counts the occurrence of a string in the source of a web page.

Here are some examples:

```
# Look for "Jake" on the specified web page
$ bash goog.sh Jake http://dsl.mwisely.xyz/labs/3/assignment/
Jake: 11

# Look for "the" on the specified web page
```

```
$ bash goog.sh the http://dsl.mwisely.xyz/labs/3/assignment/
the: 43

# Look for "The" on the specified web page
$ bash goog.sh The http://dsl.mwisely.xyz/labs/3/assignment/
The: 2

# Look for "cake" on the specified web page
$ bash goog.sh cake http://dsl.mwisely.xyz/labs/3/assignment/
cake: 0

# Give it the wrong number of arguments to see the usage
$ bash goog.sh
Usage goog.sh WORD WEBSITE
```

Behavior:

- Your script always takes exactly 2 arguments:
  - The string to search for
  - The URL of the website were looking at
- If the user misuses your script, it should show them the usage.
- The program must print the number of times the word appears in the web pages source (case sensitive!)

Hints:

- You'll want an if statement to check the number of arguments
- Use `exit NUM` to exit the shell script and return `NUM` to the shell.
- You should use pipes to redirect output
- The following commands may be useful:
  - `wget` downloads webpages.
    - The `-O` flag can be used to direct downloaded content to standard out.
    - The `--quiet` flag suppresses the download progress.
  - `grep` searches for occurrences of a string pattern
    - The `-o` flag prints each match on its own line.
  - `wc` counts lines, words, and characters
    - The `-l` flag just prints the number of lines.

Problem 3: Big Trouble in Little Whitespace

For this problem, follow the directions and write your answers in a file named `answers.txt`.

1. Use `compiley.sh` to compile `program.cpp` into an executable named `hello`.
   (a) What is the command you ran in order to compile `program.cpp` to `hello` using `compiley.sh`?
   (b) Briefly describe how the script works in plain English. (You dont need to explain the echos.)
2. Rename your program to `my program.cpp` by running `mv program.cpp "my program.cpp"`. You can run `ls -l` to make sure your file name has that space in it.
   (a) Can you still compile your program with `compiley.sh`?

   ```
   # Don't forget to escape the space when you run the script!
   $ bash compiley.sh my\ program.cpp hello

   # Or, you could use quotes
   $ bash compiley.sh "my program.cpp" hello
   ```

   (b) Based on the output and `g++` error messages, what is the problem?
3. Change the last line of `compiley.sh` to `compile_file '$@'` and try compiling your program again.
   (a) Does `compiley.sh` work now?
   (b) Whats the problem this time?
4. Change the last line of `compiley.sh` to `compile_file "$@"` and try compiling your program again.
   (a) Does `compiley.sh` work now?
   (b) Why did the double quotes (") fix the problem?

Epilogue

As with lab 2, your git repo on git.mst.edu is your submission. Don't forget to `git add` all the files you want to submit, `git commit` them, and `git push` your changes so the grader can download them!

Your repo should contain the following files:

- `README.md`

- `compiley.sh`
- `program.cpp`
- `menu.sh`
- `goog.sh`
- `answers.txt`