

CS 279 - Homework 5

Deadline:

Due by 11:59 pm on **FRIDAY**, October 5.

How to submit:

Submit your files using `~st10/279submit` on nrs-labs, with a homework number of 5, by the deadline shown above.

Purpose

To practice with some filename expansion patterns, some regular expression patterns, some looping, some quoting and backquoting, some creation of aliases and local variables, some path variations, and more...

Important notes:

- It is possible that your answers may be collected and posted to the course Moodle site.

The Problems:

Problem 1:

In a file `hw5-1.txt`, include:

- your name
- the part you are giving an answer for
- the command (or commands linked together by pipes, all in one command line) that does each of the following:

1 part a

Output the names of all files in the current directory whose names start with an uppercase X or Y or Z and end with a lowercase a or b or c.

1 part b

Output the names of all files in the current directory whose names:

- start with `test`
- end with either 0, 2, 4, 6, or 8 directly before an ending suffix `.txt`

(For example, `test2.txt` should match; `test3.txt` should not; `test12.txt` should match; `test2.txt` should not; and `oldtest2.txt` should not. Please ask me if you need more clarification on what should be matched here...)

1 part c

When you give the `grep` command a regular expression and then a single file name, it outputs the lines in that file that contain the given pattern.

Display all lines in file `fred` that include a string starting with an uppercase `X` and ending with a lowercase `a`

1 part d

Display all lines in file `fred` that include a string of 5 or more lowercase `ks` in a row immediately followed by a lowercase `j`

1 part e

Display all lines in file `fred` that include a string of 5 lowercase `ks` in a row followed by any characters and then followed by a lowercase `j`

1 part f

Add to/extend your executable path within your current shell to include `~st10/279stuff` and `~/bin` and the current directory.

1 part g

Within your current shell, create a custom version of the `rm` command so that it runs with the `-i` option whenever you type just `rm`

1 part h

The `ls` command includes an option `-t` that outputs files in order of last modified date/time, and option `-l` [<-- that's a DIGIT one, not a letter ell...] that outputs file names 1 per line.

Within your current shell, create a custom command named `lta` that lists the files, including invisible files, 1 per line, in order of last modified date/time -- but it displays them 1 screenful at a time.

Submit your resulting `hw5-1.txt`.

Problem 2:

You can put text "around" a variable's value -- you can put text beforehand by simply putting it before, and you can put text after IF you write the variable as `${myVar}` instead of `$myVar`.

For example:

```
$ pig=oink
$ echo looky$pig
lookyoink
$ echo ${pig}looky
oinklooky
```

In a file `hw5-2.txt`, include:

- your name
- the part you are giving an answer for
- the command that does each of the following:

2 part a

Create a local variable `prob2` with a value of your choice.

2 part b

Write an `echo` command to display `$prob2` has the value followed by the value of `$prob2` (note: I literally want to see a dollar sign followed by `prob2` at the beginning of this `echo` command's output)

2 part c

Write an `echo` command to display a string of 3 letter o's immediately followed by the value of `$prob2`

2 part d

Write an `echo` command to display the value of `$prob2` immediately followed by a string of 3 letter o's

2 part e

Write an `echo` command to display a string of 3 letter o's immediately followed by the value of `$prob2` immediately followed by a string of 3 letter o's

Submit your resulting `hw5-2.txt`.

Problem 3:

Write a bash shell script `gen-fodder.sh` that meets the following specifications:

- include a descriptive opening comment block including your name and the last modified date
- use a "count-controlled"/"traditional" `for`-loop to create in the current directory the 20 files `test1.txt`, `test2.txt`, ... `test20.txt`,

- each containing one line containing the file's name,
- and one line containing any line of text you'd like that also includes the file's number (test1.txt includes a 1 in its second line, test2.txt includes a 2 in its second line, etc.)

Then, demonstrate its use as follows:

- create an empty new directory
- in that directory, run:
 - echo an "about to start test" message into a file gen-test.txt (if you would like to append a blank line or some kind of border after this, you may do so)
 - append the results of an ls command to gen-test.txt to show the directory's initial contents (just gen-test.txt, at this point, and *maybe* gen-fodder.sh, although you could also call it from another directory)
- now run gen-fodder.sh in that directory
- and afterwards, echo an "after test" message and append it to gen-test.txt (if you would like to append a blank line or some kind of border after this, you may do so)
- and then append the results of another ls command to gen-test.txt to show the directory's resulting contents (should be considerably more files now)

Submit your resulting gen-fodder.sh and gen-test.txt.

Problem 4:

Write a bash shell script use-fodder.sh that meets the following specifications:

- include a descriptive opening comment block including your name and the last modified date
- use a "list-controlled"/"for-in" for-loop that uses a back-quoted command with a filename expansion pattern that will list the names of ONLY the file names that start with test and end with an even number before the suffix .txt -- that is, test2.txt, test4.txt, ... test10.txt ... test20.txt.
- for each of these selected "even numbered" files, append the line "TAG gotcha" to the end of these files

To demonstrate this script, do the following:

- go to the directory you used to test gen-fodder.sh in Problem 3 -- it should still contain the files test1.txt, test2.txt, ... test20.txt
- echo "directory contents:" into a file use-test.txt (if you would like to append a blank line or some kind of border after this, you may do so)
- list the files currently in this directory, appending the result to use-test.txt
- run use-fodder.sh in this directory
- echo "after ran use-fodder.sh:", appending the result to use-test.txt (if you would like to append a blank line or some kind of border after this, you may do so)

- then, run:

```
grep "TAG gotcha" *
```

...except append the output to `use-test.txt`

Submit your resulting `use-fodder.sh` and `use-test.txt`.

Problem 5:

`wc` is a simple UNIX filter that gives the number of lines, the number of "words", and the number of characters in a file given as its command-line argument (or in whatever it passed to it via standard input).

Write a bash shell script `txt-stats.sh` that meets the following specifications:

- include a descriptive opening comment block including your name and the last modified date
- it will output the number of lines, the number of "words", and the number of characters in each of the files whose name ends with `.txt` in the current working directory.

To demonstrate this script, do the following:

- go to the directory you used to test `gen-fodder.sh` in Problem 3 -- it should still contain the files `test1.txt`, `test2.txt`, ... `test20.txt`
- `echo "directory contents:"` into a file `txt-test.txt` (if you would like to append a blank line or some kind of border after this, you may do so)
- list the files currently in this directory, appending the result to `txt-test.txt`
- `echo "about to run txt-stats.sh:"`, appending the result to `txt-test.txt` (if you would like to append a blank line or some kind of border after this, you may do so)
- run `txt-stats.sh` in this directory, appending its output to `txt-test.txt`

Submit your resulting `txt-stats.sh` and `txt-test.txt`.