CS 279 - Exam 2 Review Suggestions - Fall 2014

last modified: 10-29-2014

- You are responsible for material covered in class sessions and homeworks; but, here's a quick overview of especially important material.
- You are permitted to bring into the exam a single piece of paper (8.5" by 11") on which you have **handwritten** whatever you wish on one or both sides. This paper must include your name, it must be handwritten by you, and it will **not** be returned.
  - Other than this piece of paper, the exam is closed-note, closed-book, and closed-computer.
- This exam will be similar in style to Exam 1, although most of the questions will focus on "new" material (material covered since Exam 1). However, concepts from Exam 1 will still be involved -- we have necessarily been building on earlier material.
- Remember that UNIX is case-sensitive; an answer may lose points if it uses the wrong case (`VI` and `Vi` are not legal UNIX commands, although `vi` is).
- Your studying should include careful study of posted examples and notes as well as the homeworks (and posted example solutions) thus far.

regular expressions

- You are responsible for being able to read, write, and understand both basic regular expressions (BREs) and extended regular expressions (EREs)
- You should be able to use regular expressions with the `grep` command, within a shell script with the `=~` operator, and within `sed` scripts
- you were responsible for just simpler uses of `grep` on Exam 1 -- you should be comfortable with more in-depth use of it on this exam.

more bash shell features - especially good for interactive shells

- backquoting/command substitution
- environment variables and local variables
  - what is the scope/lifetime of each?
  - what environment variables does a subshell have? how does it get them? what is the impact on the calling shell if they are changed in a subshell?
- what does the `source` command do? why might one choose to use it?
- initialization files
  - how can you create command aliases in an initialization file? You should be able to read, write, and understand `alias` commands
- how can you set your bash command prompt?

- what does the `PATH` environment variable contain? How is it used? You should know how to reset this environment variable.

more bash shell features - especially good for bash shell scripts

- you should be able to read, write and understand both styles of `for` loops discussed (both "list-style" and "traditional")
- you should be able to read, write, and understand `while` loops
  - how can you read the lines from a file with a `while` loop?
- you should be able to read, write, and understand `if` statements
- you should be able to write a variety of tests (as can be used in `while` loops and `if` statements)
- how can you evaluate an arithmetic expression?
- you should be able to set and use local variables in bash shell scripts
- you should be able to write bash shell scripts that make use of command-line arguments
  - how can you find out the number of command-line arguments?
  - how can you get all of the command-line arguments? ...each individual command-line argument?
- how can you do interactive input in a bash shell script?
- how can you exit a shell script at a specific point? what is the significance of the value you exit with? how can you obtain the exit status of the previous command?
- you should be able to read, write, and understand bash arrays
  - you should be able to create an array, add array elements, modify array elements, access array elements
  - you should know how to find out how many elements are in an array
  - you should be able to obtain all of the indices for an array; you should be able to obtain all of the elements in an array
- you should be able to use the `BASH_REMATCH` array to obtain what matched a subexpression in a previous regular expression; you should be able to use it to obtain what matched that regular expression as well

more file-related topics and commands

- what do the commands `basename` and `dirname` return?
- what are device files? what is special about `/dev/null`? How is `/dev/null` often used?
- you should understand and be comfortable calling the following commands; you should know how they can be useful, when you might use them, and their effects and/or output when called:
  - `cmp`

  - `diff` (including: what is the difference between `cmp` and `diff`?)
  - `wc`
  - `touch`
  - `gzip, gunzip`
  - `tar`
  - `head`
  - `tail`
  - (and a few more `ls` command options)

more commands

- you should understand and be comfortable calling the following commands; you should know how they can be useful, when you might use them, and their effects and/or output when called:
  - `which`
  - `tee`
- you should be able to understand, read, and write `sed` commands involving basic substitution (either substituting for the first instance of a pattern in a line or for all instances of a pattern); more extensive use of `sed` and other `sed` commands will be final exam fodder.

standard files and redirection

- what is meant by the standard files standard input, standard output, and standard error?
- how can you redirect each?

escaping special characters, and quoting

- how can you escape special characters on the bash command line? within a command in a bash shell script?
- what is the difference between using single quotes and using double quotes when a shell variable is involved?

the `find` command

- you should be able to read, write, and understand how to use the `find` command
- criteria you should be comfortable with include `-name`, `-print`, `-type`, `-mtime`, `-size`
- in a number of find's criteria, what is the difference between giving a number such as `3`, `+3`, or `-3`?