Intro to Linux

...

Linux

- Is an operating system (like iOS, Windows, etc...)
- Traditionally runs from a *command line*
  - Typeface interaction instead of point-and-click
  - Also called the *shell*

Shell Commands

- http://linuxcommand.org/lc3_learning_the_shell.php

- We'll go through this together...scroll down to next slide for essential information that you need to know for tests and practical purposes

Practical Commands (aka you need to know these in order to code in Linux)

- Organization
  - pwd, ls, cd, file_names
- Viewing
  - less
- File manipulation
  - cp, mv, rm, mkdir
- General commands
  - man

Organization

>> pwd

- “Print working directory”
- Folders in Linux are called *directories*

- pwd: *prints* the name of your current directory to the screen

Organization

>> ls

- Lists the files and directories within your current working directory
- Can add *flags* to it
  - *Flag* = options
    - Represented by a "*-option*" after the command
  - >> ls -a : *lists all* files (including hidden ones)
  - >> ls -l : *lists* all files in *long* form

Organization

>> cd

- “Change directory”

- To *change* into a folder: >> cd *folder_name*
- To *change* out of a folder: >> cd ..
- To *change* to your home folder: >> cd ~

Organization

File/directory Names

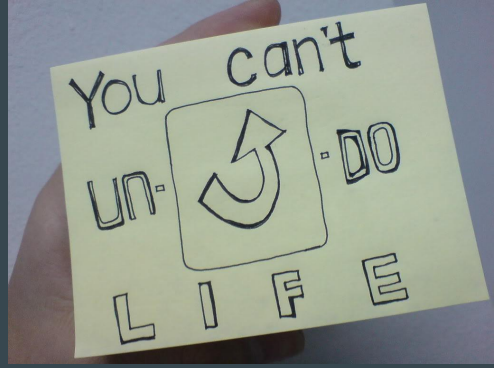
- File names CANNOT HAVE SPACES in Linux
  - FileName = Good
  - File_Name = Good
  - File.Name = Good (but bad style)
  - File Name = Bad

Organization

File/directory Names

- File names cannot start with a period
  - .FileName = Bad
  - *Technically, this is okay, but the file just won't appear in Linux*
- File names are case sensitive
  - FileName & Filename are two different files

Directories



>> mkdir, rmdir

- *mkdir <dir_name>* : makes a directory named *dir_name*
  - >> *mkdir Programming*
- *rmdir <dir_name>* : removes the directory named *dir_name*
  - >> *rmdir Programming*
  - WATCH OUT: There is no undo button for rmdir

Making Files

- Use text editors (like MS Word, Google Docs, Notability, etc...)
- Most popular
  - Nano
  - Emacs
  - VI
  - Eclipse

Making Files

>> emacs <file_name>.txt

- Opens the *emacs* text editor
- Allows you to write to a file named <file_name>.txt
- Emacs commands
  - Ctrl^X Ctrl^S to save
  - Ctrl^X Ctrl^C to save & exit
  - More commands can be found here:
  - https://www.cs.colostate.edu/helpdocs/emacs.html

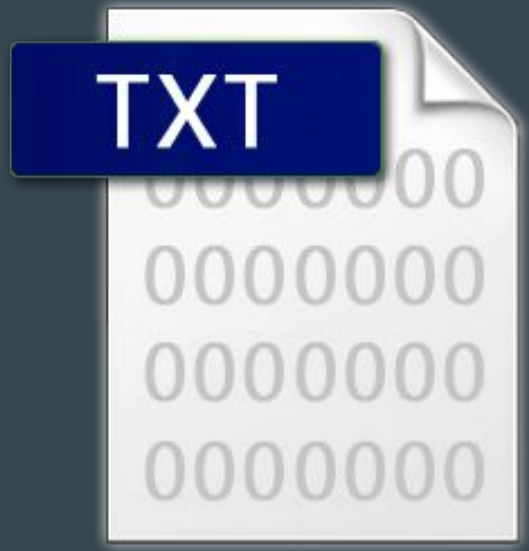
Making Files

>> nano <file_name>.txt

- Opens the *nano* text editor
- Allows you to write to a file named <file_name>.txt
- Nano commands
  - Ctrl^O to save
  - Ctrol^X to exit
  - More commands can be found here: http://staffwww.fullcoll.edu/sedwards/Nano/UsefulNanoKeyCommands.html

Making Files

- File extensions explain what kind of file you make
  - .txt ⇒ plain text file
  - .c ⇒ C file
  - .py ⇒ Python file
  - .csv ⇒ Comma separated file (Excel)



Viewing Files

>> less

- Allows you to see the contents of a file, one page at a time
- Once you are viewing a file...
  - *b* : move *back* up one page
  - *space bar* : move *down* one page
  - *q* : exit view

Files

>> cp <OG_file> <new_file>

- Copies an original file into a new file (has to be named differently than any other file in the current directory)

- >> cp test.c new_test.c
  - Copies the contents of *test.c* into *new_test.c*

Files

>> mv <file> <new_file>

- Used to *rename* a file / *move* it to a different location
- >> mv test.c final_test.c
  - *Renames “test.c” as “final_test.c”*
- >> mv test.c ../../test.c
  - *Moves test.c up two levels, makes a new file there called “test.c” with the same information as the original*

Files

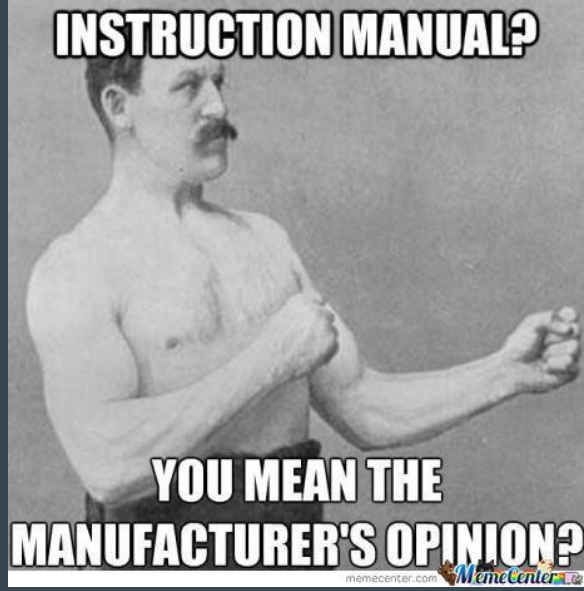
- >> rm <file>
  - *Removes* a file
  - Same as *rmdir* - but removes files instead of directories
    - No undo button
    - No prompting
  - Ex: >> rm test.c

General

>> man

- Gives the *manual* of a specific command
- Describes proper usage, flags, etc...

- Ex: >> man ls



General

>> echo

- Displays (*echoes)* the output of a command to the command line
- >> echo programming is awesome
  - Will *echo* “programming is awesome”

Test Commands (aka you need to know these for a test - and because they're cool and they will help you)

- Wildcards (http://linuxcommand.org/lc3_lts0050.php)
  - Especially * and ?

Wildcards

>> echo *

- Wildcards are characters that represent more than one character
- * ⇒ Any character
  - >> echo * : Displays everything
  - >> echo D* : Displays everything starting with the letter D, then followed by any character

Wildcards

>> ls ???????

- ? ⇒ Any *single* character
- >> ls ??????? : Displays any file/directory that is 7 characters long
- >> ls tes? : Displays any file/directory that begins with *tes* and has a 4-character name

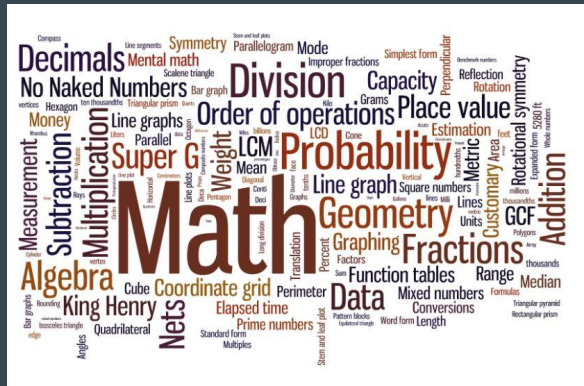
Wildcards

- Used for making shell scripts easier
  - Example: You want to list only files that end in *.c*
    - >> ls *.c
  - Example: Need to copy all files that being with “1” and are 3 characters long
    - >> cp 1?? <destination>

Shell Math

- Can also use the “echo” command to perform mathematical operations

```
>> echo $((math_expression_here))
```



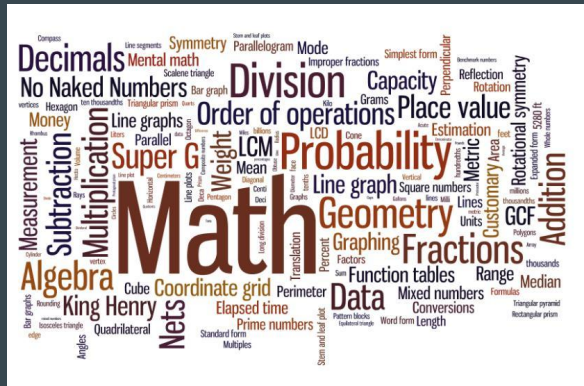
Shell Math

- Can also use the “echo” command to perform mathematical operations

- Addition

  >> echo $((5+4)))

  Can also combine commands...



Shell Math

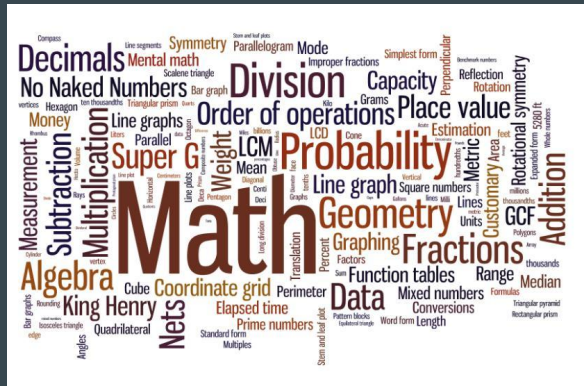
- Can also use the “echo” command to perform mathematical operations

- Addition

  >> echo $((5+4)))

  Can also combine commands...

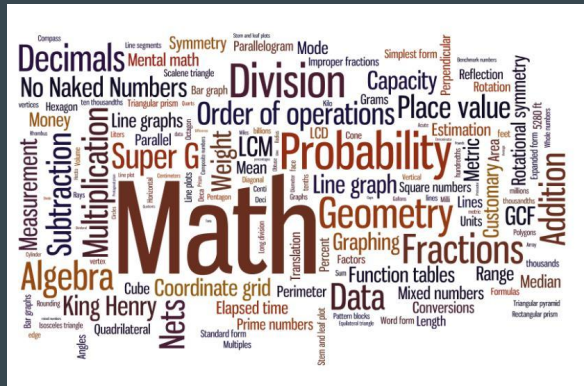
  >> echo The sum of 5 and 4 is $((5+4))



Shell Math

- Can also use the “echo” command to perform mathematical operations

- Can use any other mathematical operation
  - Subtraction: -
  - Multiplication: *
  - Division: /
  - Exponents: **



Other commands (aka you don't need to know these, but they will make you appear like a legitimate programmer)

- Expansions (http://linuxcommand.org/lc3_lts0080.php)
  - Especially what will print to the screen when brace expansion, double quotes and backslashes are used

- I/O redirection (http://linuxcommand.org/lc3_lts0070.php)
  - Especially piping (|) and manipulating standard output (>)