Unix Terminal Intro

File structures, navigating, and editing

Tanner Murphey – Astronomy Coding Education

Unix File Tree

Linux and Mac (mostly)

- Unix: everything is a file
- Directories (folders)
- Root (/): base of all files in computer
  - Contains home, OS, other important system files
  - Don't touch unless you know what you're doing
- Home (~): everything you actually interact with
  - Documents, pictures, code, etc.
- Subdirectories (subfolders)

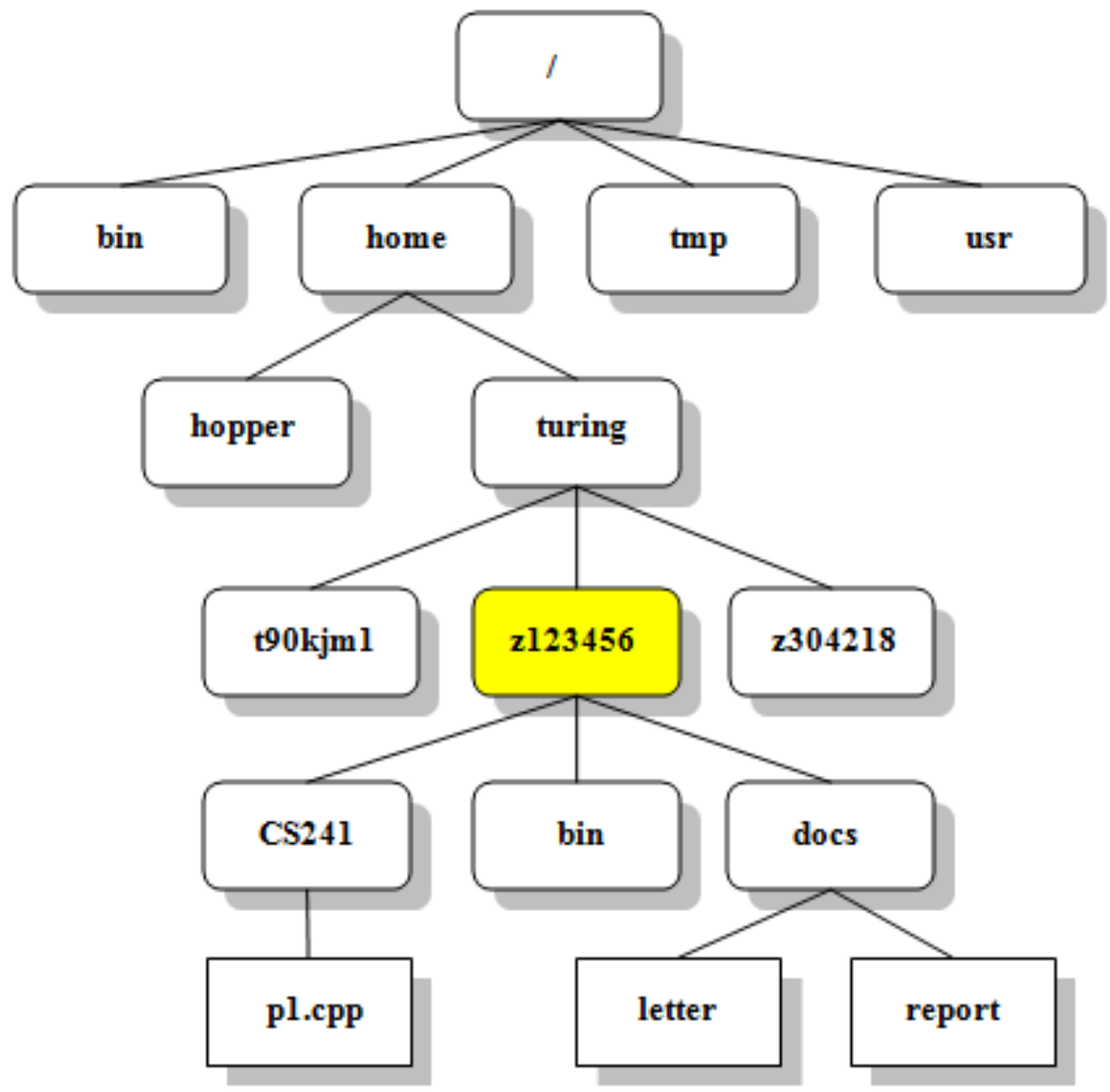

Image credit: cs.niu.edu

What is a terminal?

(Technically a terminal emulator)

- “Swiss army knife” for computers
- Can do/fix just about anything from it
- Edit files, run code, launch programs, download from internet, access remote computers, etc.
- Command-line interface (No mice 🐀)
- Shells: program that runs other programs (in the terminal in this case)
  - Bash: old standard, default on many linux distros and campus cluster
  - Z shell (zsh): modern version of bash, default on Mac, some Linux
  - Other shells can be downloaded/installed if desired (ex: fish)

Basic Commands

pwd, cd [destination]

“Print working directory”, “Change directory”

- pwd: show what directory you’re in
- cd: change to different directory
- Absolute path: full path through subdirectories from root
- From home directory: ~
- Relative path
  - Current directory (.)
  - Above directory (..)

ls [flags] [args]

List directory contents

- -l (long)
  - List much more information next to each item
- -a (all)
  - Include hidden files (start with ".")
- -r (reverse)
  - Reverse alphanumeric order
- Flags can be combined together (ex: -la)

cp, mv, rm

Copy, move, remove (files and directories)

- cp, mv: [command] [source(s)] [destination]
- rm [files]
- -r (recursive)
  - For directories, all contents within
- -f (force)
  - Operate on protected files without asking
- -v (verbose)
  - Print out every file the operation is being performed on
- Note: operations can't be undone directly
  - Undo cp -> rm <new files>
  - Undo mv -> mv <new location> <old location>
  - Undo rm -> 😬

Expansions

Get all instances of matched contents

- `?`: any single character
  - `?`, `??`, `???`, etc.
- `*`: any number of characters
- `[…]`: any single character/ range within
- `{…}`: multiple characters within (comma separated)
- `[!…]`,`{!…}`: **not** of contents within
- `!!`: previous command

Creating Items

- mkdir [name]: <u>m</u>a<u>k</u>e new <u>dir</u>ectory
  - -p: create path for new directory if doesn't exist
- touch [filename]: creates empty file if doesn't exist
  - If file exists, changes last modified date
- CLI editors (next slide)

Editing files

(Cue arguments over what's best)

- Three main editors:
  - Vim/Neovim: lightweight, fast once over learning curve, steep learning curve though
    - 3 "modes": Normal, insert, block
  - Emacs: larger than Vim, but more extensible/ usable outside of file editing, also large learning curve
  - Nano: lightest, simplest to use, but fewest capabilities
    - Only really usable for small text files

Slightly More Advanced

Displaying Contents

View files without editing

- cat <filename>: print entire contents of file to screen
- head/tail <filename>: display first/last lines of file
  - Default is 10, use -n to change
- more/less <filenames>: open page viewer in terminal of filenames

grep

Searching file contents

- grep <flags> <pattern> <file(s)>
- Outputs file name, line with matched pattern
- -r: search recursively within directory
- -v: search **not** of pattern
- -b/a/c <#>: print # lines before/after/both the matched pattern
- -i: ignore case
- Can use wildcards, regex
- Related but more advanced: sed, awk

Outputs and Pipes

Chain commands together easily

- echo [contents]: prints value of contents
- [command 1] | [command 2]: use output of command 1 as argument for command 2
- [command] > [file]: write output of command to file instead of printing to screen
  - Overwrites file if exists already
  - >> appends instead of overwrites

Sudo

"SUperuser DO"

- Some commands require admin-level privileges
  - Editing core files, installing programs, etc.
- sudo <command>
- Prompts for password
  - Will not show characters being typed for security
- USE ONLY WHEN NECESSARY
  - Used both to prevent easily messing up system files and to keep malware out

Aliases and Variables

Simplifying long commands

- Aliases
  - Shorten longer commands to simple keyword
  - To set: alias <name>=“<command string>”
    - Note: no spaces outside “”
  - Can only be used as substitute for commands
- Variables
  - More versatile than aliases: can be placed anywhere in command line
  - To set: <name>=<value>
  - To call: $name or ${name}
  - More on these in potential future bash scripting talk (stay tuned)

rc Files

Terminal startup commands

- .bashrc, .zshrc, even .vimrc
- Set anything you want to run on startup
  - Aliases, variables, Python environments, ssh keys, etc.
- Main way to make non-defaults persist over sessions
- Different shells have different rc files
  - Most shells have same basic syntax but with extra bells and whistles

Package Managers

(Best part of using terminals imo)

- Super easily download, install, and update apps/programs from command line
- Mostly OS-dependent
  - Mac/Linux: Homebrew (requires installation)
  - Linux (built in): apt (Debian/Ubuntu/Mint), dnf (RHEL/Fedora), pacman (Arch-based)
- "sudo <package manager> install <package name>"

Cheat Sheet Part 1 (Penultimate Slide)

- clear: clear the screen
- history: list previous commands (-c to clear)
- man <command>: display manual entry for command
  - Only guaranteed for built-in commands, try -h or --help for others first
- which <command>: see exactly which file is run when using a command
  - Very useful for python environments and other similar programs

Cheat Sheet Part 2

Keyboard Shortcuts

- Ctrl+a(e): jump to beginning (end) of current line
- Ctrl+u: delete whole current line
- Ctrl+c: end current running process
- Ctrl+z: suspend/pause current running process
  - bg/fg: run suspended process in background/foreground
- Ctrl+d: stop the terminal
- Tab completion
  - Displays all possibilities if multiple available
- ↑↓: scroll through old commands