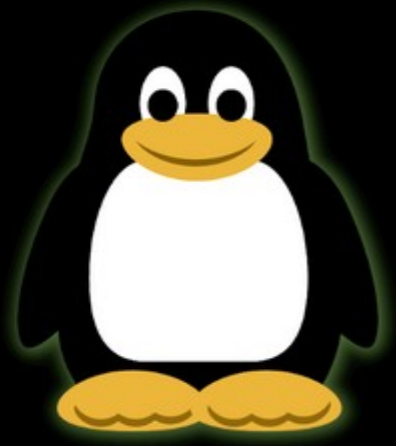
$ bash



Using Linux Lunch and Learn #2: Bash

What we'll try to cover today

- Introductions
- What is a shell?
- Bash
- Starting the shell
- The command line
- Working with directories and files
- Pipes and redirection
- Working with processes
- Question and answer session

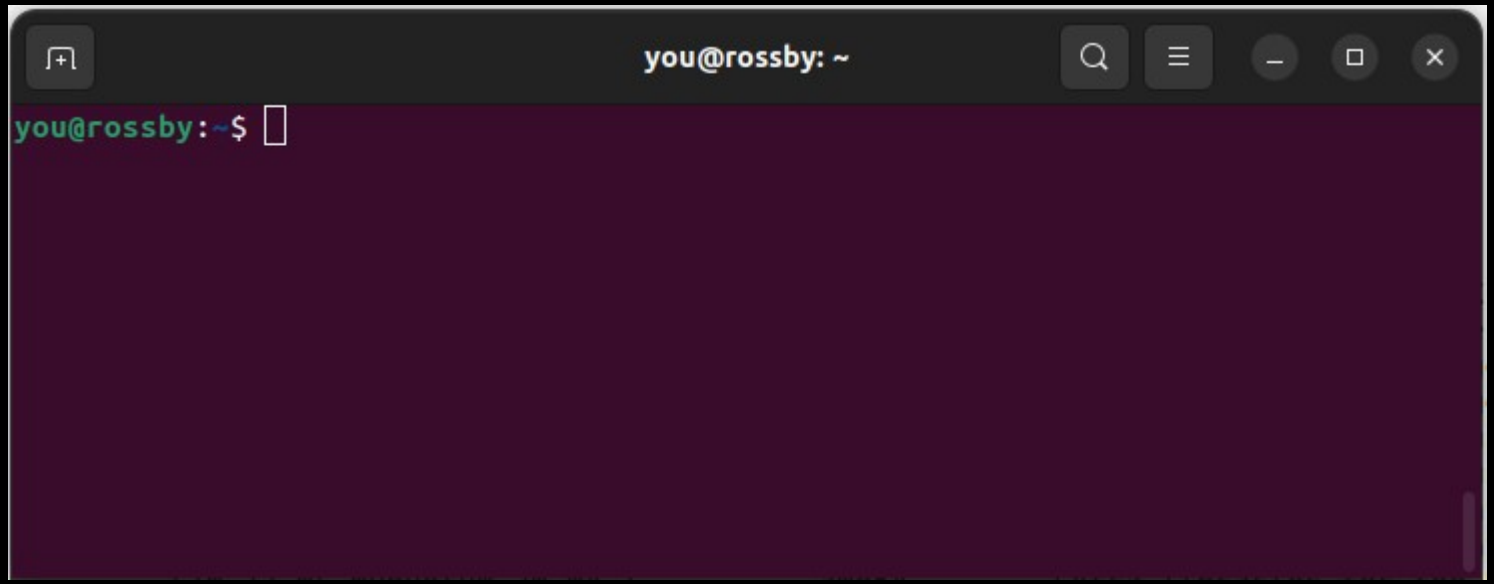
About me

- Erik Meitner
- Systems Administrator for the Math Dept.
- Room 507
- What I do
- How I can help you
- Other IT staff: Sara Nagreen and Henry Mayes

About you

- Name
- What was the best moment of the summer for you?

What is a shell?



A shell is

- An application
- A textual interface to the operating system
- An interpreter for a shell script

Shells

Some Linux shells:

- Bourne shell sh
- Almquist shell (ash)
- Debian Almquist shell (dash)
- Bash (Unix shell) bash
- KornShell ksh
- Z shell zsh
- C shell csh
- TENEX C shell tcsh
- Ch shell ch
- Emacs shell eshell
- Friendly interactive shell fish
- PowerShell pwsh
- rc shell rc, a shell for Plan 9 from Bell Labs and Unix
- Stand-alone shell sash
- Scheme Shell scsh

From: https://en.wikipedia.org/wiki/List_of_command-line_interpreters

Bash

- The default shell on all our Linux servers and workstations
- Available on Mac OS (though TCSH is the default)
- Available on Windows 10 via the Windows Subsystem for Linux
- https://www.gnu.org/software/bash/

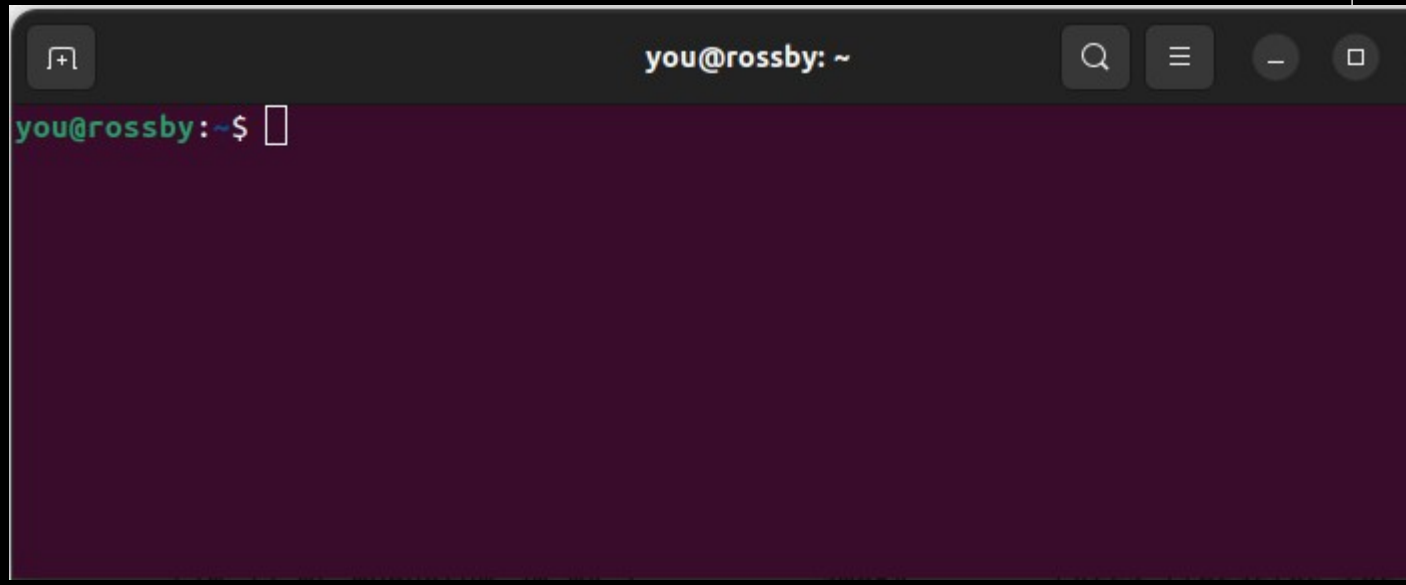
• Starting the shell

A shell starts when you:

- Open a terminal application locally
- SSH to a remote computer
- Login to a local console(Linux)

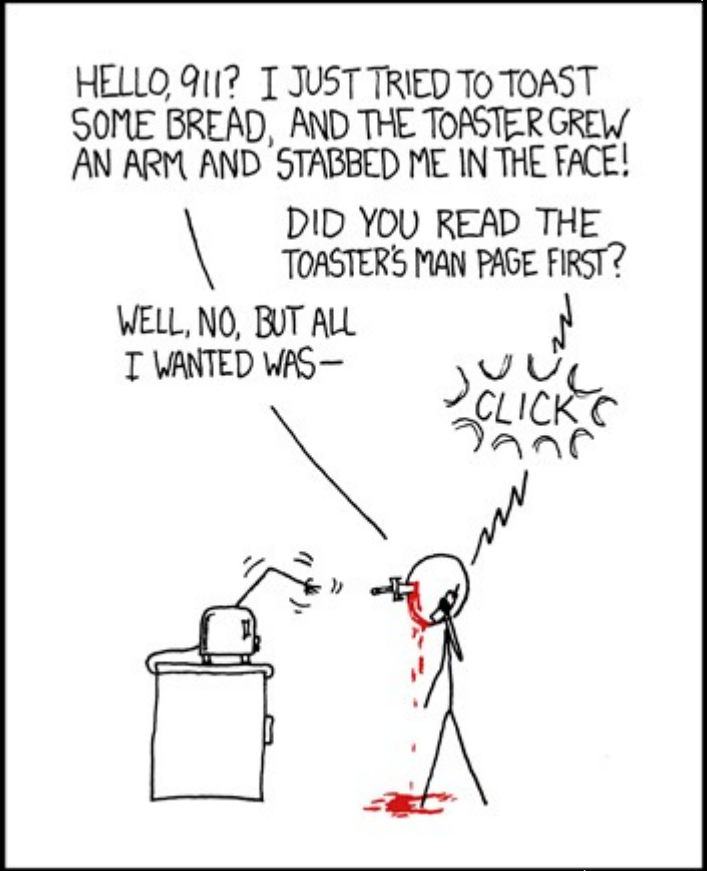
The command line

- Username
- Machine name
- Current directory
- Prompt character
  - $
  - #
- You can set your prompt to whatever you like
  - PS1='C:\> '



Man: Learn to use it

- Linux systems have built-in reference documentation.
- The command to access is is "man", short for "manual"
- search for a man page with the -k option
  - man -k edit
- View a man page:
  - man nano
- Man uses the "less" command to show the manual in a paginated way.
  - Space moves forward a page
  - "b" moves back a page
  - "q" quits
  - "/" lets you search starting at your current page
  - "n" goes to the next matching text
  - " N" goes to the prevous matching text



Side note: Filesystem heirarchy

```
In Linux EVERYTHING IS A FILE-LIKE OBJECT!
/
├── bin
├── boot
├── build
├── dev
├── etc
├── fac
    └──
├── grad
    └── yourname
├── home
├── lib
├── opt
    └── intel
├── proc
├── root
├── run
├── sbin
├── scratch
├── srv
├── staff
    └── emeitner
├── sys
├── tmp
├── usr
    └── local
        └── bin
├── var
└── visitor
```

Working with files and directories 1

- pwd - Print working directory
- ls - List files/directories
  - ls -l
  - ls -a
  - ls -lS
  - ls -lt
- cd - Change directory
- ~ - Alias for your home directory
- . - Alias for the current directory
- .. - alias for the parent directory of your current directory

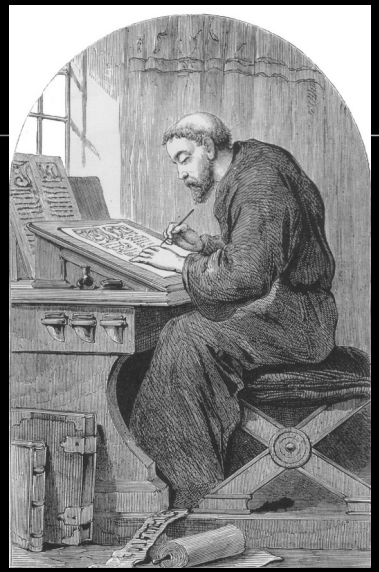
Working with files and directories 2

- mkdir - Make a directory
- rmdir - remove an empty directory
- rm - remove file/directory
  - rm -r
- cp - copy file/directory
  - cp -r
- mv - move/rename a file/directory

Working with files and directories 3

- less - view the contents of a text file
  - less fps9000.f95
- cat - send the contents of one or more files to the terminal
  - cat file1.txt fileA.txt footer.txt
- hexdump - view the contents of any file in hexadecimal
  - hexdump -C /bin/cvt

Editing text files



Many editors are available.

- nano - Easy to use
- joe - Easy to use, more complex than nano
- vi - available on all UNIX/Linux systems.
- emacs - Complex and highly extensible and programmable
- ...and hundreds more...

Pipes and redirection

- Each process has three data streams:
  - stdin - Standard input, file descriptor #0
  - stdout - Standard output, file descriptor #1
  - stderr - Standard error, file descriptor #2
- These data streams can be redirected to/from files or other processes.

Redirection

- Send the output(stdout) of a command to a file
  - ls -l > listing.txt
- Send the contents of a file to a program's input(stdin):
  - wc -l < code.c
  - But... wc -l code.c
- Send stdout to one file and error messages(stderr) to another
  - noisy_program > log.txt 2> errors.txt
  - “>” and “1>” are the same
  - "2>" means redirect stderr
- Send stdout to a file and send stderr there as well
  - test_app > /tmp/output.txt 2>&1
  - “&1” means “wherever stdout is going”
  - "2>&1" means send stderr to wherever stdout is going
- Send stdout to a file but append to the file rather than overwriting it
  - tool1 >> logs/tool1.log 2>&1

A few more commands: grep

- grep - This one would require an entire class to cover all it’s uses. I strongly suggest you get to know this commands basic functions.
- grep needle haystack.txt
  - looks for the text “needle” in the file haystack.txt
- grep -i dx1 libkrud.h
  - looks for “dx1” in libkrud.h, ignoring case
- grep -r lost_it code/
  - looks for files containing “lost_it” in the code directory recursively
- Note that by default grep uses a syntax called Regular Expressions(RE) in the search term. Some characters have special meaning. Unless you understand RE syntax it is best to use the -F option and always use quotes around the search term.
- grep -F '\right|^s' paper.tex
  - In this case "\r" would have tried to match the carriage-return character(ASCII 13). The -F option prevents that.

A few more commands: find

- Find looks for files.
- find .
  - prints out every file and folder in the current directory, and all directories it contains
- find ~ -name my_keys.txt
  - Looks for a file in your home directory named my_keys.txt
- find /opt/intel/ -mtime -60 -iname "*diffcalc*" -type f
  - Finds all files(not directories) in /opt/intel/ modified in the last 60 days that contain "diffcalc" in their name

Pipes 1

- Pipes connect the stdout of one program to the stdin of another
- ls /dev/ | less
- find /usr/local/matlab/ | fgrep .F
- grep -ri 'sub doThing' repo/grass/

• Working with processes 1

- w - show who is logged in now
- last -10 - show last 10 users to log in
- ps - list running processes
- kill - stop a process
- top - watch a continuously updating sorted list of processes
- nice - starts a process at a non-default priority
- renice - changes the priority of a running process

• Working with processes: w and last

- w
- last -10
  - Show last 10 people to log in
- who
  - similar to w

• Working with processes: ps

- ps
  - list your processes that were started by your current shell
- ps -f
  - same but with more details
- ps -ef
  - show all system processes in with more detail
- ps -U emeitner
  - Shows ALL my processes
- ps -ef | grep -F parallel_test
  - Lists all processes and matches lines that contain "parallel_test"
- ps -U kfrog | grep -F demo1 | wc -l
  - Lists all processes run by user kfrog, matches those lines containing demo1, the counts the number of lines

• Working with processes: kill

- kill 342342
  - kill process ID 342342 using the default TERM(terminate) signal
- kill 342342 45433 34533
  - kill a number of processes
- Hm... PID 342342 didn't stop.
- Kill -INT 342342
  - Try the INT signal(interrupt)
- Hmm.. still running.
- kill -HUP 342342
  - Try the HUP(hangup) signal
- Ok. Process ID 342342 is broken. The code should be fixed. But for now lets just stop it at all costs - even data loss
- kill -KILL 342342

• Working with processes: top

- Shows you whats consuming the most system resources. By default it shows CPU usage.
- "q" to quit
- "h" for help, "ESC" to exit help
- PID - Process ID
- USER - Owning user
- PR - Priority of the process assigned by the Linux kernel
- NI - Niceness. A user alterable priority. Values from -20 to 19. Lower is "nicer"
- VIRT - The total amount of virtual memory used by the task.(KiB) It includes all code, data and shared libraries
- RES - Resident Memory Size (KiB) A subset of the virtual address space (VIRT) representing the non-swapped physical memory a task is currently using.
- SHR - Shared Memory Size (KiB)
- S - Status. D = uninterruptible sleep, I = idle, R = running, S = sleeping, T = stopped by job control signal, t = stopped by debugger during trace, Z = zombie
- %CPU - The task's share of the elapsed CPU time since the last screen update, expressed as a percentage of total CPU time. By default it shows % of CPU time used on the single CPU core a process is using.
- %MEM - A task's currently resident share of available physical memory
- TIME+ - Total CPU time the task has used since it started.
- COMMAND - The program being run by the process

• Working with processes: nice/renice

- All user processes start with a niceness of 0 by default.
- nice my_app
  - runs my_app at a niceness of 10
- nice -n 19 your_app
  - runs your_app with a niceness of 19(lowest priority)
- renice -n 5 -p 2345334
  - Sets niceness of running process with PID 2345334 to 5

Some helpful links

- Bash reference manual, https://www.gnu.org/software/bash/manual/html_node/index.html
- Linux command line for you and me, https://lym.readthedocs.io/en/latest/index.html
-

Notes from today

- Will be posted on the Math Dept. wiki: https://wiki.math.wisc.edu/
- Search for “lunch and learn”

Next time

- Time, date, and topic to be announced on the mailing list
- To join the list send an email to: math-linux-help+join@g-groups.wisc.edu

Contacting me

- You can always contact me directly with questions:
  emeitner@math.wisc.edu
  608-263-4189(office)
- Or stop by my office:
  Van Vleck room 507

Thank you