CSE 374: Programming Concepts and Tools

Spring 2026
Instructor: Megan Hazen

Today's Goals

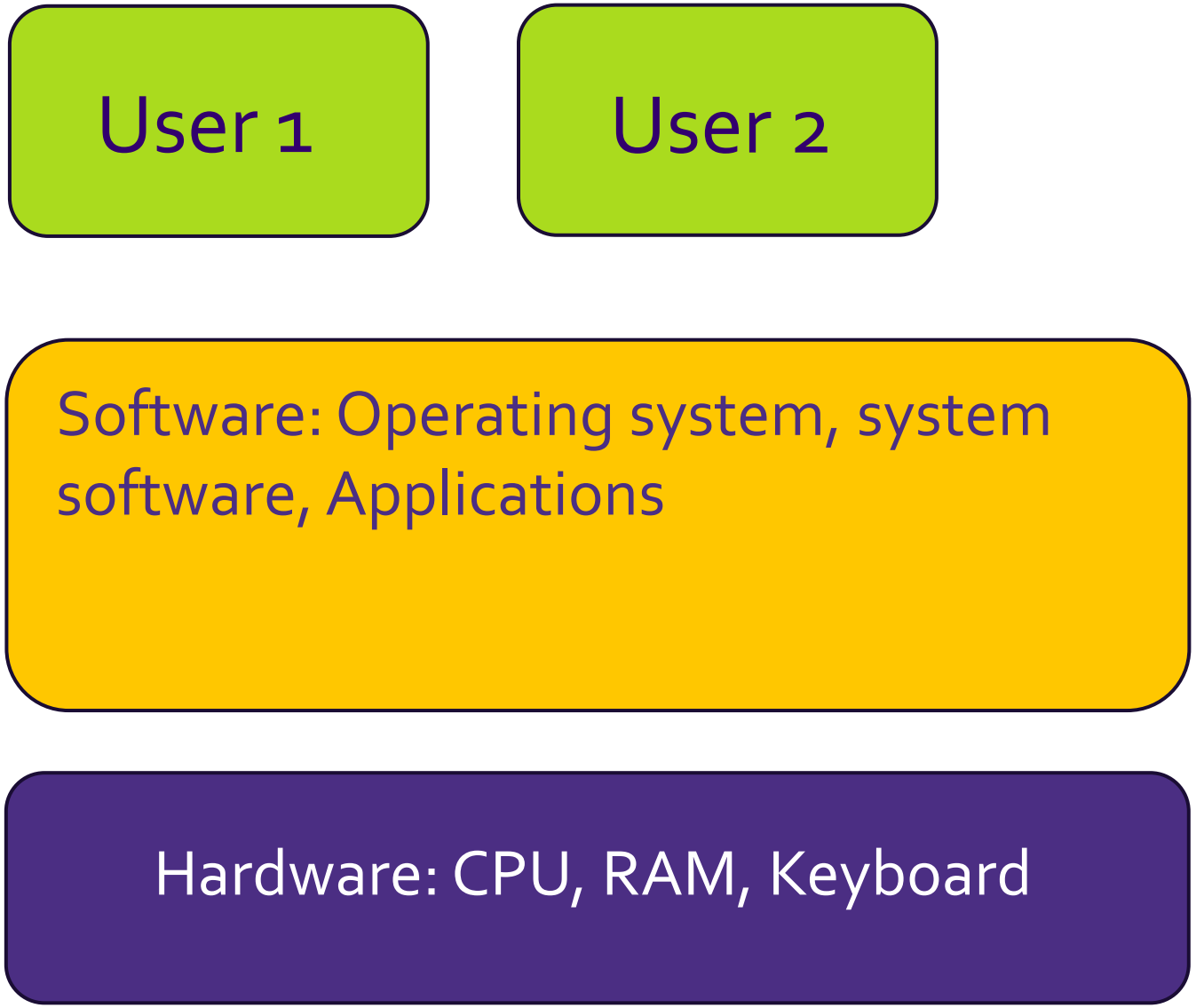
Subject Matter

- Computer Model
- What is Linux?
- What is the Shell?
  - What is Bash?
- Accessing Calgary
- Getting started with Bash

Your Goals

- Log onto Calgary
- Do first practice problem on Gradescope
- HWo
  - Find description on webage
  - Complete HWo
  - Find Gradescope to turn it in

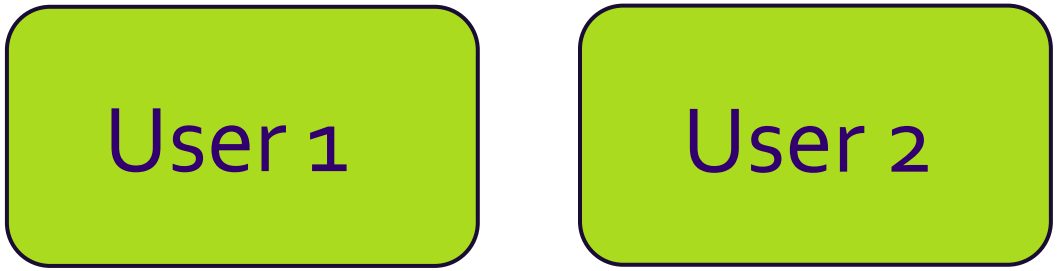
Computer Model



Hardware

- Motherboard
  - Power
  - CPU – registers to store data & instructions, logic circuits to execute instructions
  - GPU - many low powered CPUS in parallel
- RAM (Random Access Memory)
- Disk drive / solid state drive
- Keyboards & Mice
- Monitors
- Network components

Hardware: CPU, RAM, Keyboard



Users

One or more users simultaneously
Run processes (software)
Interact with hardware

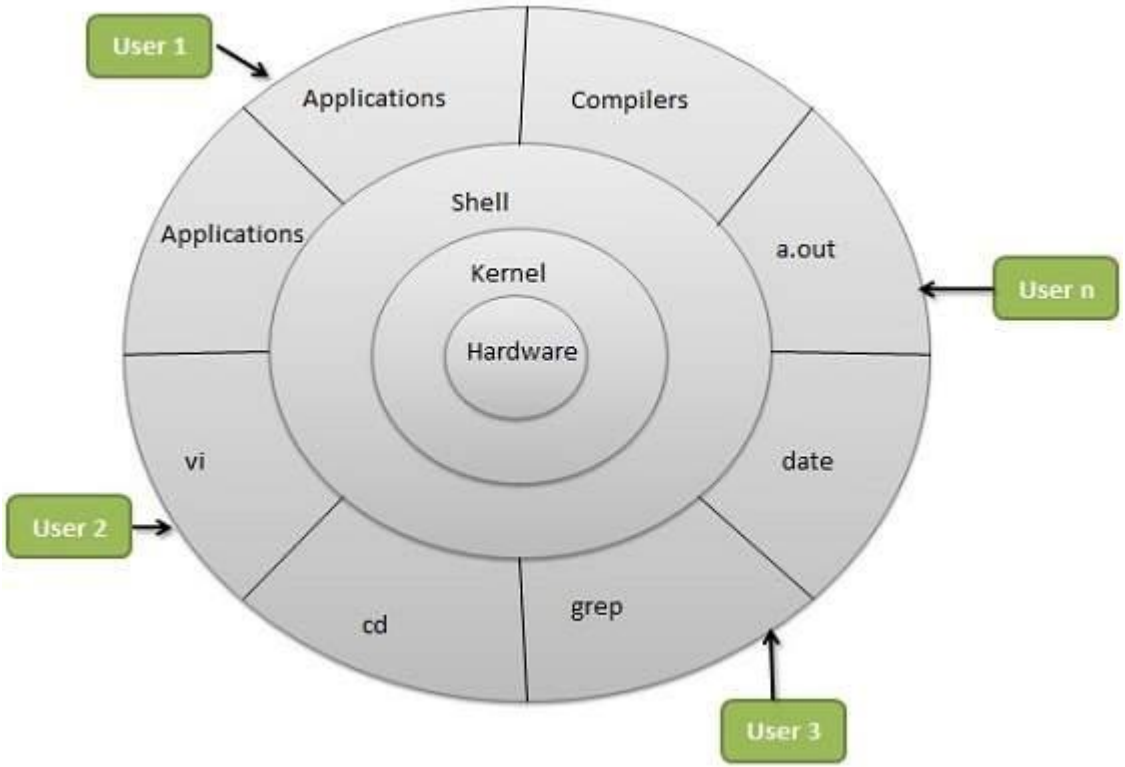
Software

Operating system
- Hardware abstraction (device drivers)
- Networking
- Scheduling
- File System

Software: Operating system, system software, Applications

System software
System libraries
Application software

What is Linux?



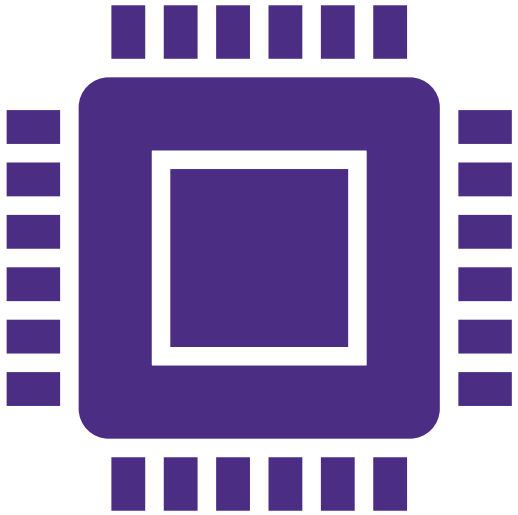
Linux is an operating system

Portable, open source, built with C

Kernel – does all the hardware interaction

Shell – provides interface for users

Users – control and run various processes (applications)



WHAT OPERATING SYSTEMS CAN YOU NAME?

Linux variants

- Different kernel versions
  - For different hardware
  - Newer features
  - Optimized for different things
- Different distributions
  - Run on a kernel
  - Offer different software on top of the kernel
- Calgary runs Rocky Linux (distro) version 10.1, 64-bit address space

```
[mh75@calgary ~]$ uname -a
Linux
calgary.cs.washington.edu
6.12.0-
124.40.1.el10_1.x86_64 #1
SMP PREEMPT_DYNAMIC Tue Mar
3 18:15:11 UTC 2026 x86_64
GNU/Linux
```

What is the shell?

- Text is efficient - typing is fast, and there aren't big image objects to pass around
- Scripting makes it easy to automate text-based interfaces
- Linux *does* have a graphical interface
- Windows and MacOS *do* have shell interfaces
- Most power users use BOTH

- This course uses Calgary, running a Rocky Linux distribution (or flavor?)
- You could use any distribution of Linux that is up-to-date, but may have inconsistencies.
- There are also 'flavors' of shells. We will use Bash for this course.
- For most things you can use any terminal interface - MacOS standardly is using Zsh

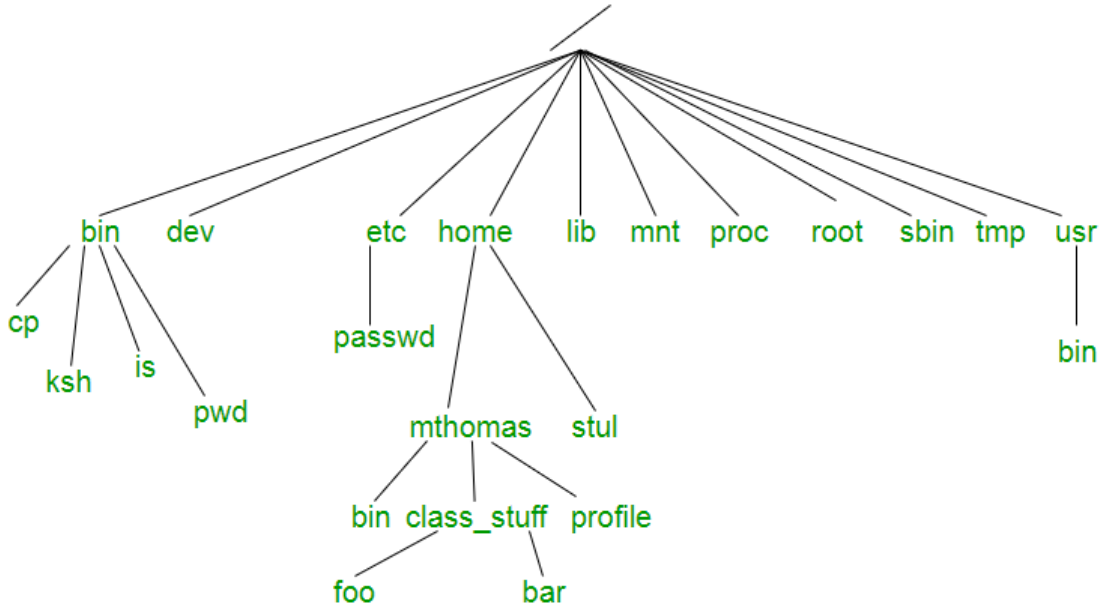
Log into Calgary

- https://courses.cs.washington.edu/courses/cse374/26sp/resources/linux.html
- *Everyone should have an account on calgary that uses your united log-in - send email to cse374-staff if you can not access yours.*

```
PS C:\Users\mh75> ssh mh75@calgary.cs.washington.edu
```

File Systems

- Data is stored in files
- Files are organized in a file system
- File systems are trees
- Absolute file address starts from the top (`/`)
- Relative file address from current position
  (`./`) – this directory
  (`../`) – one directory up
- Home directly has short-cut (`~`)



https://tldp.org/LDP/intro-linux/html/sect_03_01.html
- Linux layout, but windows is similar

Demo: `whoami, pwd, ls, mkdir, cd, cp, mv, rm, cat`

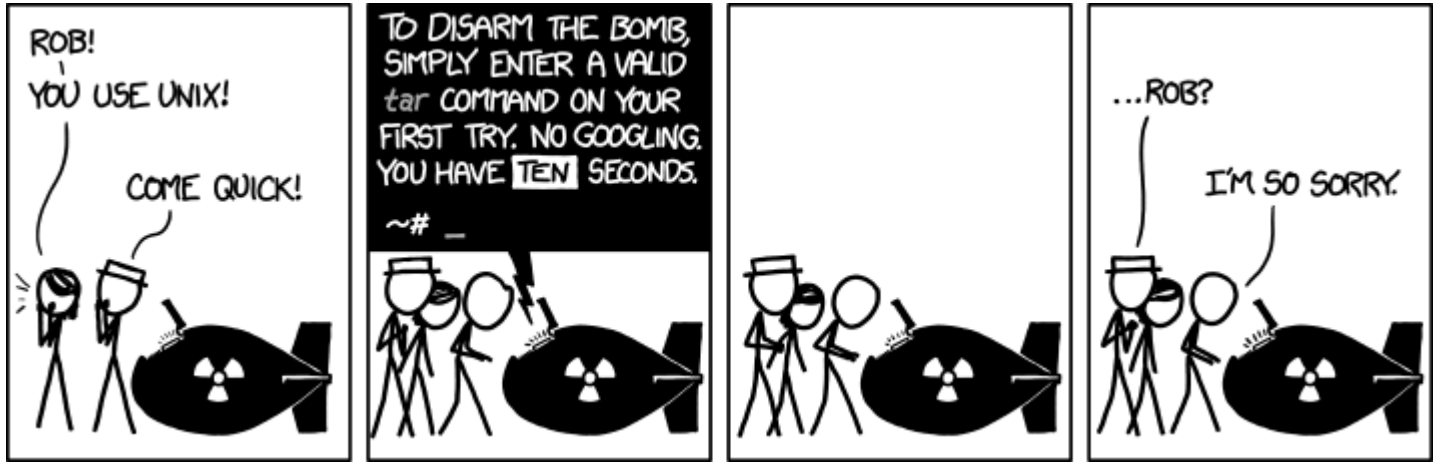
Processes

"On a UNIX system, everything is a file; if something is not a file, it is a process."

- Shell essentially runs programs, or processes. Shell *is* a process and has a state.
- Usually launch a process and return to shell when done.
- Each process has own memory stream and I/O
- Stdin (keyboard), stdout (console), stderr
- Many processes have options

- '`&`' runs process in the background
- `fg, bg, top, kill`
- Step through a script with built-in `source`
- Can redirect input and output ('`<`', '`>`', '`>>`')
- Redirect output, `cat, more/less`

Getting Help




```
[mh75@calgary ~]$ man date
[mh75@calgary ~]$ date –help
[mh75@calgary ~]$ man –k date
[mh75@calgary ~]$ apropos date
```

- https://tldp.org/LDP/intro-linux/html/sect_02_03.html

Bash as a language

Bash acts as a language interpreter

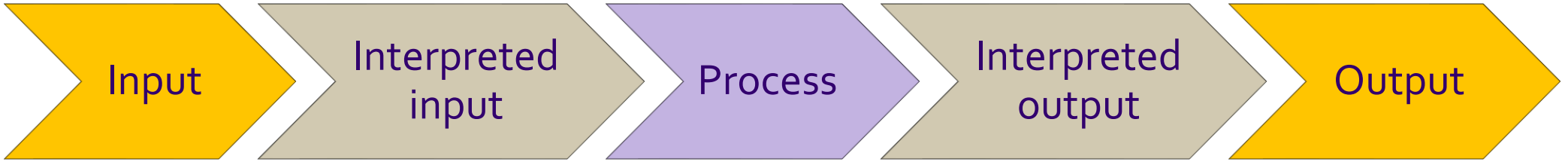
- Commands are subroutines with arguments
- Bash interprets the arguments & calls subroutine
- Bash also has its own variables and logic

Bash Globbing

Bash applies its own processing (***globbing***) to the I/O (Input / Output)

Globbing at work

Bash does redirection, string-expansion, & substitutions
Process only sees the interpreted I/O



Bash Special Characters

Directory Shortcuts
- ~uname or ~
- ./ or ../

Wildcards - Globbing
- 0 or more chars: *
- Exactly 1 char: ?
- Specified chars: [a-f]

History or `!`

! > < & | * ~ [] “
‘ ` $ /

\ is escape character

“string”
- Glob within the string

‘string’
- Use the string as-is

Office hours today

- 10:30-11:00 CSE1 462