Objectives

- Reviewing lab
- Introduction to
  - Problem solving
  - Algorithms
  - Programming languages

1

Typical Class Period Organization

1. Pearls of wisdom from Professor Sprenkle
2. Review course material in rows
   - Consult your notes, handouts, slides from recent classes (see course web site)
3. Review as a class
4. New stuff!
   - Some think-pair-share work

2

Course Logistics: Where Do I Find …?

Slides from class Example programs from class Labs Readings	→	Course web page
Zoom class recordings Discussions for Broader Issues Zoom links for office hours	→	Canvas
"Pre lab" assignments	→	Interactive Textbook

3

Course Logistics

- Handouts
  - Slide number won't always line up with slides
  - Won't always get to all
  - Don't look ahead
- Office hours this week
  - Today: 2-4 p.m.
  - Tomorrow: 10:30-11:30, 3:30-4:30 p.m.
  - Join on Zoom or stop by my office and I'll join you in the advanced lab

4

Review: Lab

- Learned some UNIX commands
- Created a Web page
- Lessons learned:
  - Problems are fixable (often just typos!)
  - No need to say you're "sorry". You're learning!
  - Learn from, adapt examples
  - Find a good solution
  - Honing your mental model

5

Review: UNIX

- UNIX is a bad parent
  - Doesn't tell you when you've done something right
  - Only tells you when you've done something wrong

Terminal:

```
sprenkle@spartacus Desktop$ cp lab00.ppt.pdf lab00.pdf
sprenkle@spartacus Desktop$
```

Did it work? Maybe.
While you're learning, need to check/confirm it!

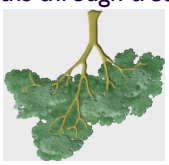
6

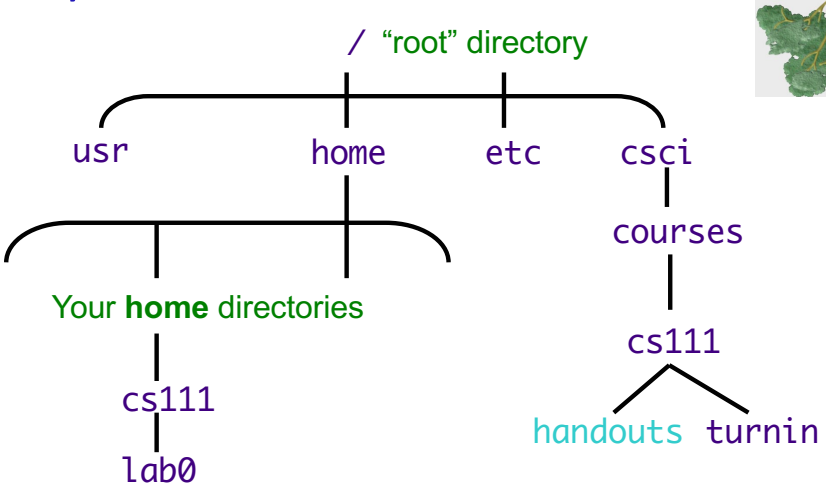
Review: Linux

- What is the syntax of the command to
  - List the files in a directory?
  - Change your current directory?
  - Make a directory?
  - Find out the current directory?
  - Make a copy of a file?
- What is the shortcut for
  - The current directory?
  - The parent directory?
  - Your home directory?
- What is the difference between an *absolute* path and a *relative* path?
  - How do you know if a path is an *absolute* or *relative* path?
- What is the *absolute path* to your home directory?
- What is an algorithm?
- What is a program?

(Partial) Linux File Structure

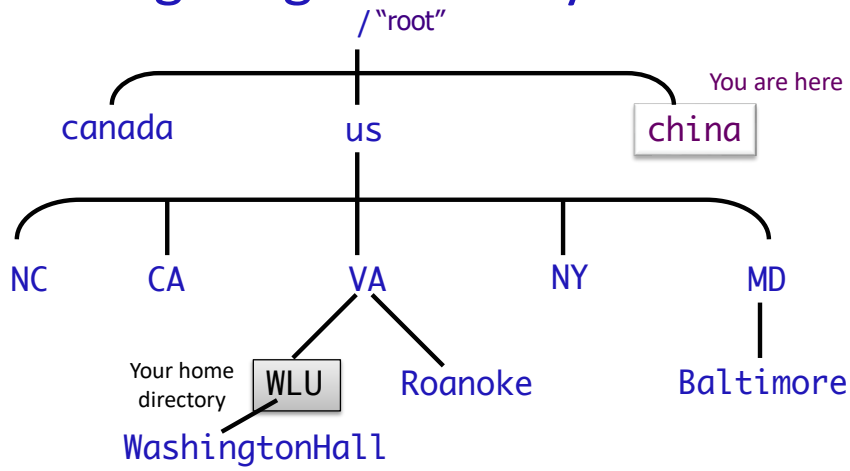
Paths through tree





What is the *absolute* path to the handouts directory?

Review: Navigating the File System



Given that you're in /china,
how would you go to canada? WLU? Washington Hall?

Linux: Helpful Trick

- If you ran a command that isn't working
  - Example: the prompt doesn't come back, and it looks like the terminal is hanging without response
  - Example: your command isn't correct

Use Control-C to stop the command

- You should get the prompt back, perhaps with a message (that probably won't make sense to you)

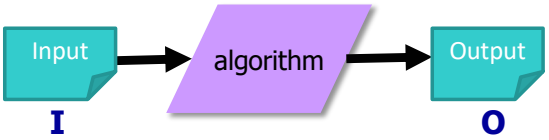
Post-Analysis of Labs

- "That's it?"
  - Often, students get overwhelmed by the directions, but then the work isn't that difficult
  - It's ***problem solving***
    - Here's what I need to do. Here's what I know.
    - How do I bridge the gap between them?
- Worth 35% of your grade
  - Should get in B+/A- range easily with help from student assistants and me

Review: Computational Problem Solving 101

- **Algorithm**: a well-defined recipe for solving a problem
  - Has a finite number of steps
  - Completes in a finite amount of time
- Program
  - An algorithm written in a programming language
  - Also called code
  - Large programs, solving many problems together → application

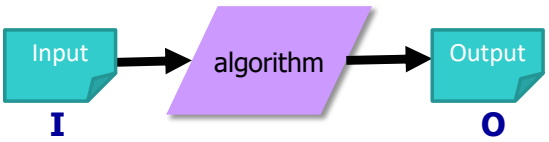
Algorithms: Input and Output



- Algorithms often have a defined **input** and **output**
- Correct algorithms give the intended output for a set of input
- Example: Multiply by 10
  - I/O for a correct algorithm:
- More examples
  - averaging numbers, recipes

Input	Output
5	50
.32	
x	

Algorithms: Input and Output



- Algorithms often have a defined **input** and **output**
- Correct algorithms give the intended output for a set of input
- Example: Multiply by 10
  - I/O for a correct algorithm:
- More examples
  - averaging numbers, recipes

Input	Output
5	50
.32	3.2
x	10x

Making a Peanut Butter & Jelly Sandwich

- How do you make a peanut butter and jelly sandwich?
- Write down the steps so that someone else can follow your instructions
  - Make no assumptions about the person's knowledge of PB&J sandwiches
  - The person has the following materials:
    - Loaf of bread, Jar of PB, Jar of Jelly
    - 2 knives, a paper plate, napkins
- Algorithm: What is the input? What is the output?

15

Discussion of PB&J

- The computer: a blessing and a curse
  - Recognize and meet the challenge!
- Be unambiguous, descriptive
  - Must be clear for the computer to understand
  - "Do what I **meant**! Not what I said!"
    - Motivates programming languages
- Creating/Implementing an algorithm
  - Break down pieces
  - Try it out
  - Revise

16

Discussion of PB&J

- Steps need to be done in a particular order
- Be prepared for special cases
  - Any other special cases we didn't discuss?
- Aren't necessarily spares in real life
  - Need to write correct algorithms!
- Reusing similar techniques
  - Do the same thing with a little twist
- Looping
  - For repeating the same action

Parts of an Algorithm

- Input, Output
- Primitive operations
  - What data you have, what you can do to the data
- Naming
  - Identify things we're using
- Sequence of operations
- Conditionals
  - Handle special cases
- Repetition/Loops
- Subroutines
  - Call, reuse similar techniques

An overview for the semester!

Other Lessons To Remember

- A cowboy's wisdom: Good judgment comes from experience
  - How can you get experience?
  - Bad judgment works every time
- Program errors can have **bad** effects
  - Prevent the bad effects--especially before you turn in your assignment!

19

Looking Ahead

- Lab 0 due Friday
- Broader Issue write up on Canvas due Thursday at 11:59 p.m.

20