Objectives

- Review Lab
- Introduction to
  - problem solving
  - programming languages
  - writing python programs

Review: Lab

- Learned some UNIX commands
- Created a Web page
- Started writing Python programs

- Lessons learned:
  - Problems are fixable, find a good solution

Review: Linux

- How do you …
  - Learn more about a Linux command?
  - List the files in a directory?
  - Change your current directory?
  - Make a directory?
  - Find out the current directory?
- What is the shortcut for …
  - The current directory?
  - The parent directory?

Review: Linux File Structure

```
/           /usr  /home  /etc
|           |       |       |
| faculty   | students | courses |
|          |         |    www |
|          |         |     tmp |
|          |         |       |
| faculty  | students | courses |
|         |         |       |
| public   | handouts | turnin |
|         |          |       |
| public   | handouts | turnin |
```

Relative Paths vs Absolute Paths

- Given that you’re at WLU, how would you get to Washington Hall? To Roanoke? To Baltimore?
- Given that you’re in China, how would you go to Canada? WLU? Washington Hall?
Computational Problem Solving 101

• Computational Problem
  ➢ A problem that can be solved by logic

• To solve the problem:
  ➢ Create a model of the problem
  ➢ Design an algorithm for solving the problem using the model
  ➢ Write a program that implements the algorithm

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

Application

• Large programs, solving many problems

More on Algorithms

- 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:

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>50</td>
</tr>
<tr>
<td>.32</td>
<td>3.2</td>
</tr>
<tr>
<td>x</td>
<td>10x</td>
</tr>
</tbody>
</table>

- More examples: averaging numbers, recipes

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, Jelly, 2 Knives, paper plates, napkins

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

Discussion of PB&J

- Be prepared for special cases
- 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

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!

Computational Problem Solving 101

- Computational Problem
  - A problem that can be solved by logic
- To solve the problem:
  - Create a model of the problem
  - Design an algorithm for solving the problem using the model
  - Write a program that implements the algorithm

Computational Problem Solving 101

- Computational Problem
  - A problem that can be solved by logic
- To solve the problem:
  - Create a model of the problem
  - Design an algorithm for solving the problem using the model
  - Write a program that implements the algorithm

Why Do We Need Programming Languages?

- Computers can't understand English
  - Too ambiguous
- Humans can't easily write machine code

Why Do We Need Programming Languages?

- Computers can't understand English
  - Too ambiguous
- Humans can't easily write machine code
Programming Languages

- Programming language:
  - Specific rules for what is and isn’t allowed
  - Must be exact
  - Computer carries out commands as they are given
- Syntax: the symbols given
- Semantics: what it means
- Example: III * IV = 3 * 4 = 12
- Programming languages are unambiguous

Python Interpreter

1. Validates Python programming language expression(s)
   - Enforces Python syntax
   - Reports syntax errors
2. Executes expression(s)
   - Runtime errors (e.g., divide by 0)
   - Semantic errors (not what you meant)

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

Printing Output

- `print` is a special command
  - Displays the result of expression(s) to the terminal
  - `print "Hello, class"`  
    - `print` automatically adds a ‘\n’ (carriage return) after it’s printed
  - `print "Your answer is", 4*4`
    - Displays same as:
      - `print "Your answer is"`, `print 4*4`

Extra Credit Opportunity

- 10 points applied to Lab grade
- Attend a CS talk, all in Parmly 405
  - Mon, Jan 12, D period
  - Thurs, Jan 15, 3:30 p.m.
    - Mark Liffiton, “Satisfying Constraints, and What To Do When You Can’t”
  - Fri, Jan 23, 4 p.m.
    - Joshua Stough, “Appearance Models for Medical Image Segmentation”
- Post summary on Sakai, following CS Issues write up
Next Time

• More programming fundamentals
• Broader Issue: Technology Education
  ➢ Post write up on Sakai, as response to appropriate topic
  ➢ Your write up will include
    • How interesting you found this article on a scale of 0 to 9
    • Summary of the 3 most important points
    • Article’s effect on your understanding of CS
    • Article’s relation to our course specifically (if applicable)
    • Question for class discussion
  ➢ See Course’s CS Issues page for more information