Objectives

- Learning Linux
  - Linux practice
- Programming practice
  - Print statements
  - Numeric operations, assignments
- Web Page

Lab Review

1. What are the names of our student assistants and tech support person?
2. What OS do the lab computers run?
3. What is the terminal?
4. What is ssh?

Log into your lab machine

Lab System Review

- Everything you do in lab on these machines (if you save it), you can access remotely (on lab machines)
- Everything you do remotely on lab machines (if you save it), you can see on the lab machines in person

Lab 0 Feedback

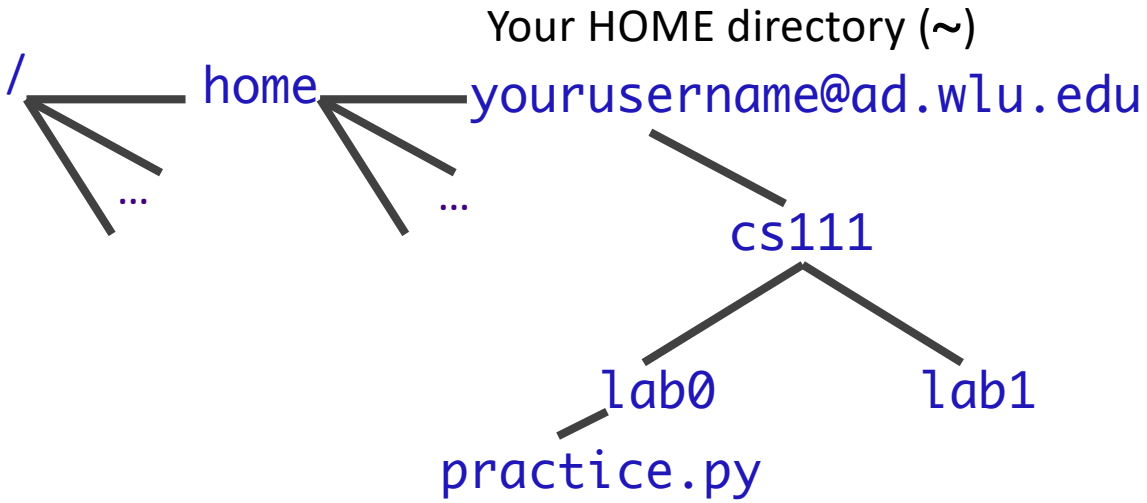
- Overall, did well
  - Generally, lab grades should be high
- Canvas extra credit Easter egg
  - Great fun facts!

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)

Review: Linux File System



~ is a shortname for your home directory, i.e., short for /home/yourusername@ad.wlu.edu

- What is the *syntax* for the copy command?
- How would you copy `practice.py` to your `lab1` directory if you were in `lab0`? If you were in `lab1`?

PYTHON PROGRAMMING

Review

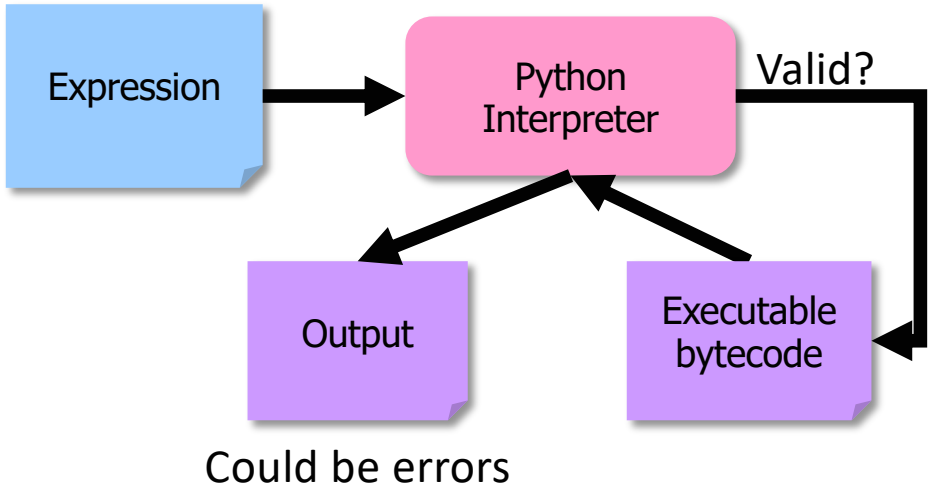
1. What are the two ways to run the Python interpreter?
2. Give three examples of primitive data types
3. How do we display output from a program?
   - What is the syntax and what does it mean?
4. How do we assign values to variables?
5. What arithmetic operators are available?
   - What rules do they follow?
6. What is our development process?

Recap: Programming Fundamentals

- Most important data types (for us, for now): `int, float, str, bool`
  - Use these types to represent various information
- Variables have identifiers, (implicit) types
  - Should have "good" names
  - Names: start with lowercase letter; can have numbers, underscores
- Assignments
  - `x = y` means "x set to value y" or "x is assigned value of y"
  - Only variable on LHS of statement changes

Review: 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*)



Review: Process of Developing Computational Solutions

1. Create a sketch of how to solve the problem (the algorithm)
2. Fill in the details in Python
3. Execute the program
4. If output doesn't match your expectation
   - Debug the program (Where is the problem? How do I fix it?)

Not necessarily complete program at first

Our development process will evolve over time

Good Development Practices

- Design the algorithm
  - Break into pieces
- Write comments FIRST for each step
  - Elaborate on what you're doing in comments when necessary
- **Implement** *and* **Test** each piece *separately*
  - Identify the best pieces to make progress
  - Iterate over each step to improve it

When to Use Comments

- Document the author, high-level description of the program at the top of the program
- Provide an outline of an algorithm
  - Separates the steps of the algorithm
- Describe difficult-to-understand code

PYTHON PROGRAMMING IN LINUX

IDLE Development Environment

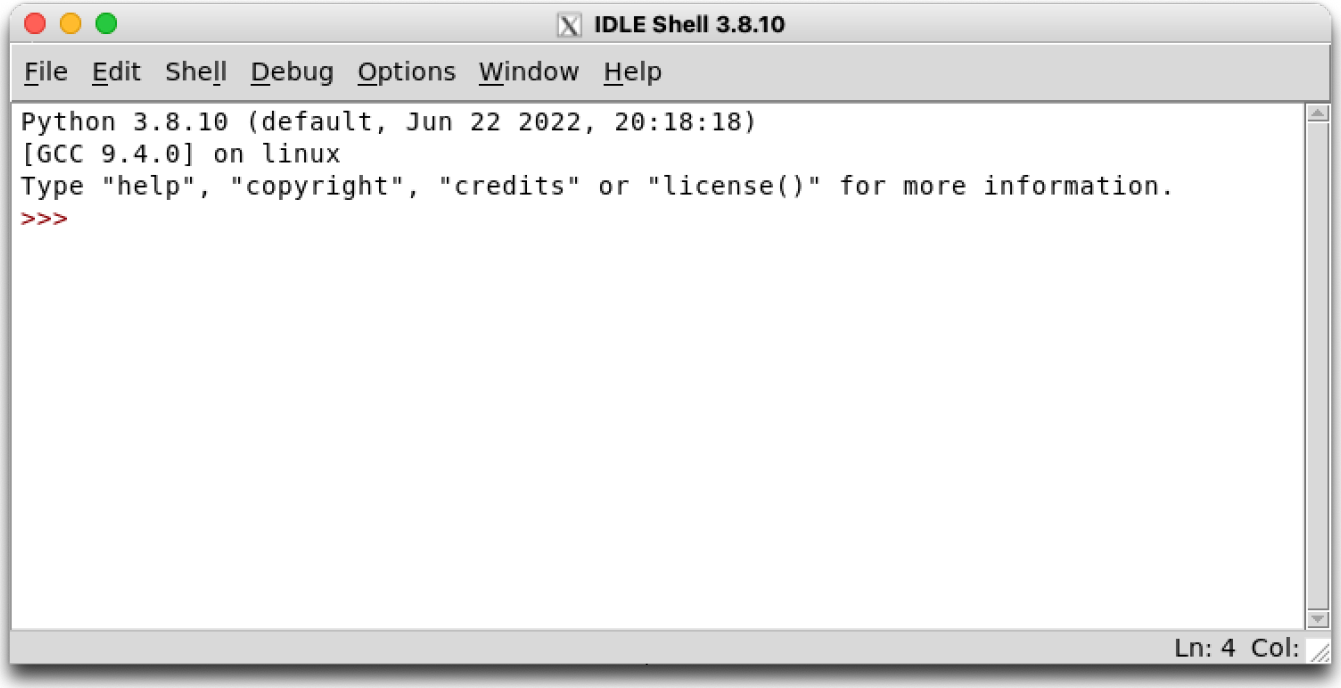
- Runs on top of Python interpreter
- Command: `idle &`
  - `&` Runs command in "background" so you can continue to use the terminal

Since our programming language is
named after Monty Python,
what is the development environment named after?

- Can use IDLE to
  - Run Python in **interactive** mode
  - Write and execute scripts in **batch** mode

IDLE

- IDLE first opens up a Python ***shell***
  - i.e., the Python interpreter in interactive mode



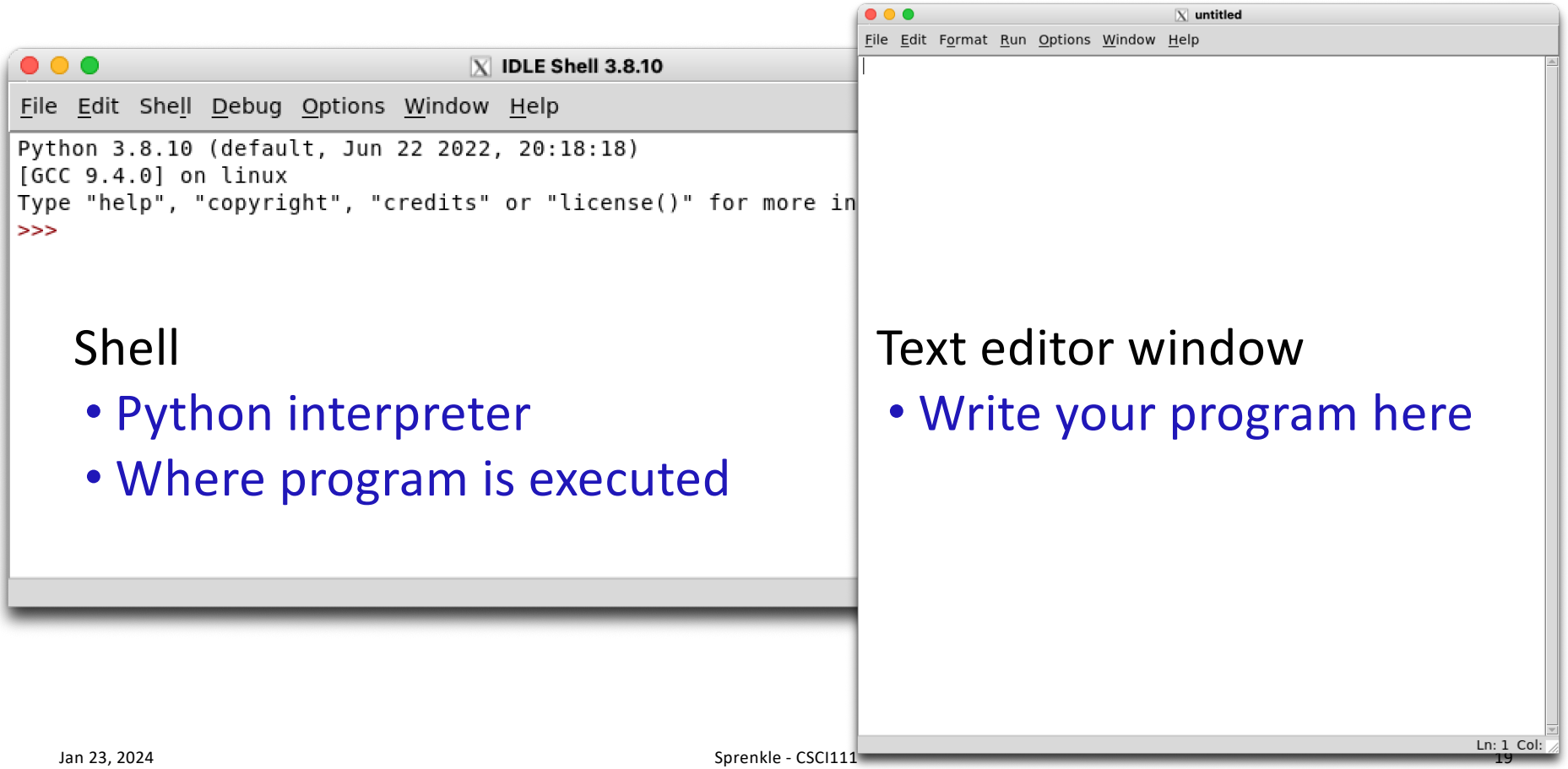
Your Turn in Interactive Mode…

- If you exited IDLE, run `idle &`
- Enter the following expressions and see what Python displays:
  - `3`
  - `4 * -2`
  - `-1+5`
  - `2 +`
  - `print("Hello!")`
- Alternatively, can use `python`
  - If you use `python`, use Control-D to quit the interpreter

Python scripts in IDLE

- In IDLE, to create a script, under the `File` menu
  - Use `New File` **or** `Open`, as appropriate, to open a window so that you can write your Python script.
- Practice:
  - Create a new file
  - Print out "hello!"
  - Save the file in your home directory
  - Execute the program (opens a new Python shell)
    - `Run → Run Module` or `F5`

IDLE: Interactive Development Environment



Shell

- Python interpreter
- Where program is executed

Text editor window

- Write your program here

Recap: Executing Python

- Interactive Mode
  - Try out expressions
  - `python`
- Batch Mode
  - Execute Python scripts
  - `python <pythonscript>`
- **IDLE** combines these two modes into one *integrated development environment* (IDE)
  - `idle &`

Lab 1 Expectations

- Comments in programs
  - High-level comments, author
  - Notes for your algorithms, implementation
- Nice, readable, clearly labeled understandable output
  - User running your program needs to **understand** what the program is saying

Lab 1: Programming Practice

- After the warm up problems…
- Name program files **lab1_n.py**, where *n* is the problem you're working on
- After completed implementation, demonstrate that your program works
  1. Close IDLE/Python interpreter, rerun program
     - Get rid of the output from when you were developing/debugging ("scratch work")
  2. Save output for each program in file named `lab1_n.out` where *n* is the problem you're working on

Lab 1 Expectations: Example Output

- Your program should have clearly labeled output
  - Clear to user what is happening in program

- Resulting output should be saved in a `.out` file

Development Process for Lab

- Develop in **IDLE**

1. Create a new file
2. Develop the program
3. Close the shell
4. Run the program again
5. Save output from program

Making a Web Page

- Leftover from last week
- Goals:
  - Practice using Linux, ssh, text editor, following examples
  - Set up for a future lab

Lab 1 Expectations: Read the Directions

- To ***completion***
- Often the answer to your question is in the next sentence
- Practice patience
  - Rushing →poor outcomes

Lab 1 Submission

- Electronic
  - I can execute your program, help find mistakes
  - Copy your lab directory into your `turnin` directory
  - And your web page!
- Printed
  - So I can provide written feedback
- Instructions are in the lab

Honor

- You may discuss programming assignments *informally* with other students/people
  - Sharing the **code** is an honor violation
  - Do **not** share your password
- You should know where to draw the line between legitimate outside assistance with course material and outright cheating
  - Students who obtain too much assistance without learning the material ultimately cheat themselves
- If you have any uncertainty about what this means, consult with me before you collaborate.

Honor System: Rules of Thumb

- Discussion of problems/programs - OK
  - Clarification questions
  - Algorithm discussion (on paper, board)
- Do ***not*** look at another student's solution
  - "What did you do for that?"
- Debugging help
  - Programmer always "owns" keyboard, mouse
  - Helper can read other's program/debug/help, up to 5 minutes
    - Ask student assistant or me or email me for problems that require more time

Honor System: AI

- You should not use AI-based assistants (e.g., ChatGPT or GitHub Co-Pilot)
- You will not have access to AI-based assistants on exams
- We are learning to problem solve and the fundamentals of programming
  - After you have practiced enough, you can be faster than the computer/would be able to leverage an AI-based assistant better

Lab 1 Overview

- Linux practice
- IDLE practice
- Programming practice
- Web page