http://www.cs.cornell.edu/courses/cs1110/2021sp

# Lecture 1: Introduction, Types & Expressions

(Chapter 1)

CS 1110

Introduction to Computing Using Python

[E. Andersen, A. Bracy, D. Fan, D. Gries, L. Lee, S. Marschner, and W. White]

## CS 1110 Spring 2021: Announcements

http://www.cs.cornell.edu/courses/cs1110/2021sp

**Sections**

- Please attend only the Section in which you are enrolled
- Use Student Center to change (swap) section if necessary: 201-205 are in-person sections; 206-217 are online sections

**Enrollment**

- A lot of turnover in the first week:  don't give up!
- Perhaps another class meets your needs?

http://www.cs.cornell.edu/courses/cs1110/2021sp/alternatives.html

**AEW Workshops** (ENGRG 1010) Open to **all** students.

Additional (optional) discussion course. Small group, collaborative learning. Non-remedial. Highly recommended.

http://www.cs.cornell.edu/courses/cs1110/2021sp/aew.html

## Why learn to program?

(subtly distinct from, although a core part of, CS / IS)

*Like philosophy … **computing is worth teaching** less for the subject matter itself and more **for the habits of mind that studying it encourages.***

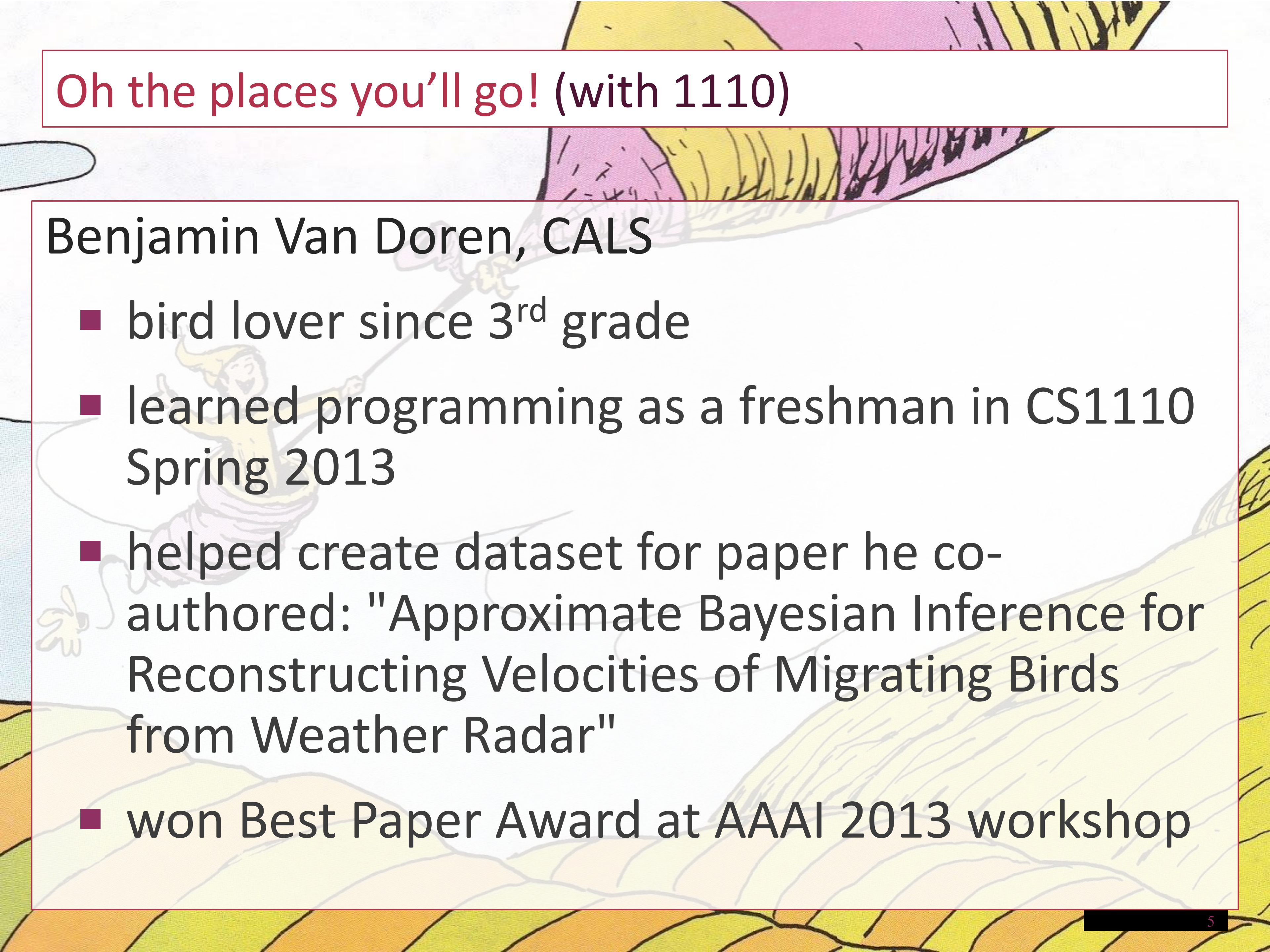
*"Teach computing, not Word"*, the Economist

http://www.economist.com/blogs/babbage/2010/08/computing_schools

## **Why learn to program?** (continued)

[T]he seductive intellectual core of… programming: here is a magic black box. [T]ell it to do whatever you want, within a certain set of rules, and it will do it; within the confines of the box you are more or less God, your powers limited only by your imagination. But the price of that power is strict discipline: you have to *really know* what you want, and you have to be able to express it clearly in a formal, structured way that leaves no room for the fuzzy thinking and ambiguity found everywhere else in life…

**…The ability to make the machine dance to any tune you care to play is thrilling.**



## Oh the places you'll go! (with 1110)

Benjamin Van Doren, CALS

- bird lover since 3rd grade
- learned programming as a freshman in CS1110 Spring 2013
- helped create dataset for paper he co-authored: "Approximate Bayesian Inference for Reconstructing Velocities of Migrating Birds from Weather Radar"
- won Best Paper Award at AAAI 2013 workshop

## About Professor Lee

Research lifetime achievement awards:

- Association for Computing Machinery (ACM), 2018
- Assoc. for the Advancement of Artificial Intelligence (AAAI), 2013
- Assoc. for Computational Linguistics, 2017

In the press: New York Times, All Things Considered, Washington Post, etc.

Engineering teaching awards: 1999, 2002, 2012

Carpenter Memorial Advising Award: 2009

A.B. Cornell ’93, Ph.D. Harvard ’97

Lowest grade ever…?

## About Professor Fan

- Interest in optimization—what is the "best" way to operate a system given constraints and uncertainties?
- Other courses:
  - Intro to computing using Matlab
  - Optimization with metaheuristics

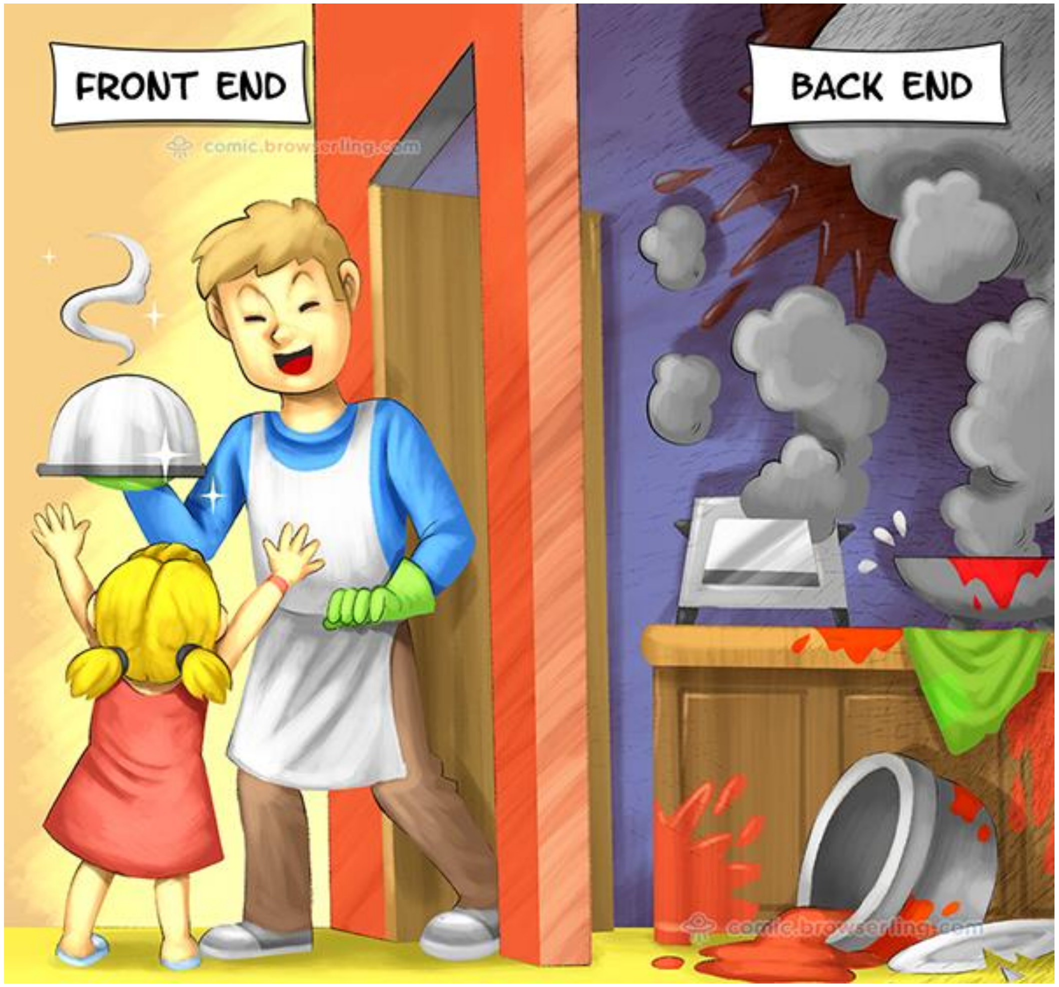
Source: energy.gov

- Author: *Insight Through Computing: A Matlab Introduction to Computational Science and Engineering* with C. F. Van Loan
- Honors:

  National Academy of Engineering Frontiers of Engineering Education (2014)
  Carpenter Memorial Advising Award (2016)
  Engineering teaching awards (2011, 2019)

# Who does what?

What you see: What you don’t see:



http://www.catonmat.net/blog/front-end-vs-back-end-comic/

## Why should you take CS 1110?

**Outcomes:**

- **Fluency:** (Python) procedural programming
  - Use assignments, conditionals, & loops
  - Create Python modules & programs
- **Competency:** object-oriented programming
  - Recognize and use objects and classes
- **Knowledge:** searching & sorting algorithms

# Intro Programming Classes Compared (1)

## CS 1110: Python

- No programming experience necessary
- No calculus
- Non-numerical problems
- More about software design

## CS 1112: MATLAB

- No programming experience necessary
- 1 semester of calculus
- Engineering-type problems
- More about computational science & engineering

Both serve as a pre-requisite to CS 2110

# Intro Programming Classes Compared (2)

## CS 1133: Python Short Course

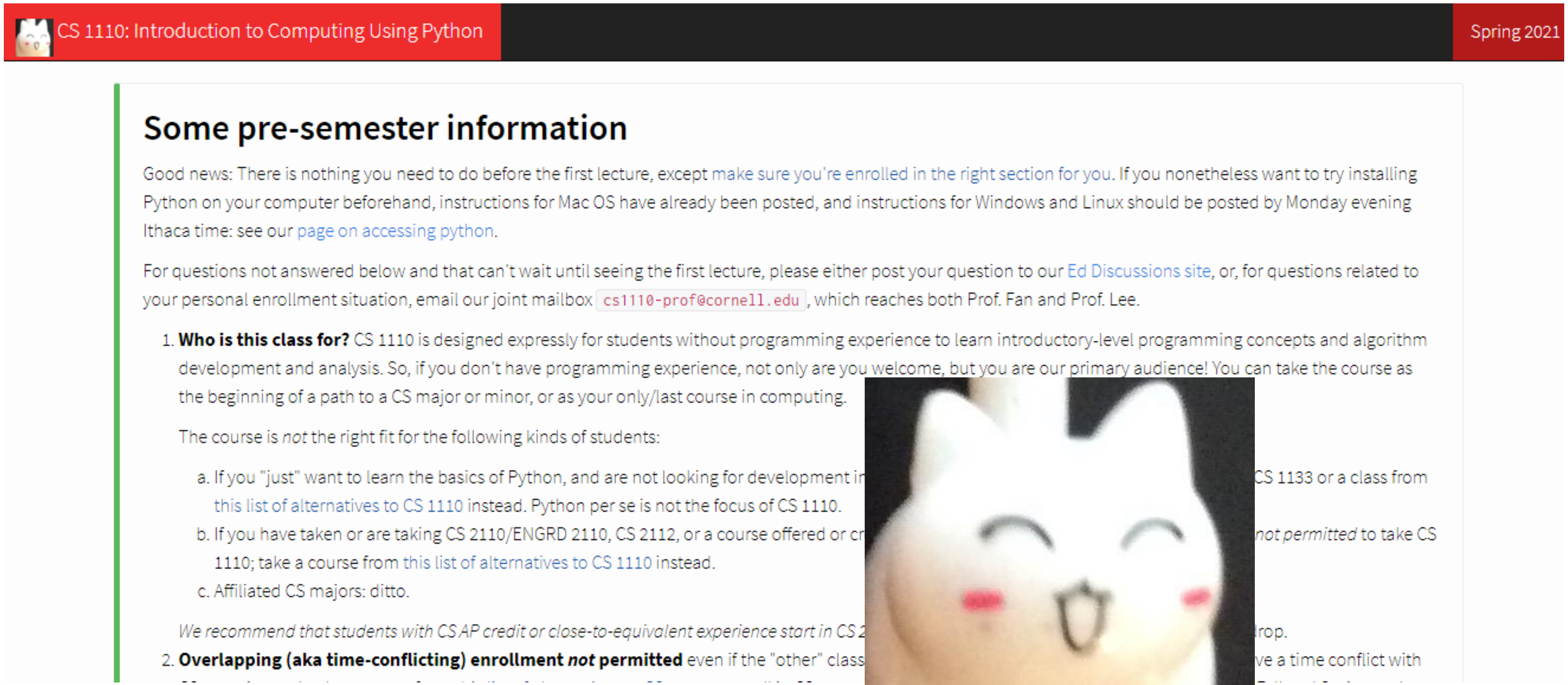
- No programming experience necessary
- No calculus
- Very basics of programming
- 2 credits (7 weeks)

## CS 1380: Data Science For All

- No programming experience necessary
- No calculus
- Less programming than 1110, but also: data visualization, prediction, machine learning

# Course Website

http://www.cs.cornell.edu/courses/cs1110/2021sp/



CS 1110: Introduction to Computing Using Python

Spring 2021

## Some pre-semester information

Good news: There is nothing you need to do before the first lecture, except make sure you're enrolled in the right section for you. If you nonetheless want to try installing Python on your computer beforehand, instructions for Mac OS have already been posted, and instructions for Windows and Linux should be posted by Monday evening Ithaca time: see our page on accessing python.

For questions not answered below and that can't wait until seeing the first lecture, please either post your question to our Ed Discussions site, or, for questions related to your personal enrollment situation, email our joint mailbox `cs1110-prof@cornell.edu`, which reaches both Prof. Fan and Prof. Lee.

1. **Who is this class for?** CS 1110 is designed expressly for students without programming experience to learn introductory-level programming concepts and algorithm development and analysis. So, if you don't have programming experience, not only are you welcome, but you are our primary audience! You can take the course as the beginning of a path to a CS major or minor, or as your only/last course in computing.

   The course is *not* the right fit for the following kinds of students:

   a. If you "just" want to learn the basics of Python, and are not looking for development i CS 1133 or a class from this list of alternatives to CS 1110 instead. Python per se is not the focus of CS 1110.
   b. If you have taken or are taking CS 2110/ENGRD 2110, CS 2112, or a course offered or cr *not permitted* to take CS 1110; take a course from this list of alternatives to CS 1110 instead.
   c. Affiliated CS majors: ditto.

   *We recommend that students with CS AP credit or close-to-equivalent experience start in CS 2* rop.
2. **Overlapping (aka time-conflicting) enrollment *not* permitted** even if the "other" class ve a time conflict with

If the website doesn't look like this, with the white cat logo, at the top left, you're looking at the wrong semester.

## Lectures

- Tuesday & Thursday 9:05am
- Not just talking! Demos, clicker questions, *etc.*
- ***Watch pre-lecture videos ("lessons") or read from supplemental textbook*** *before class!* Posted on course website the day before class. Lecture assumes that you have done the pre-lecture viewing/reading
- Lecture slides, code examples, and lecture recording available on website later, within 24 hours
- Watch the lessons and attend (or watch recording of) lecture regularly—don't get behind

## Lab (aka Sections)

- Guided exercises with TAs & consultants
- Start today: Tuesday, Feb 9
- **Attend the lab section in which you are enrolled.** We can't maintain workable staff/student ratios otherwise.
  - Need a different Section? Change (swap) section on Student Center
- Each lab has 2 parts, released on Tuesday: Part A due on Fri; Part B due the next Tues
- **Mandatory**. Missing > 4 parts (equivalent to 2 full labs) can lower your final grade.

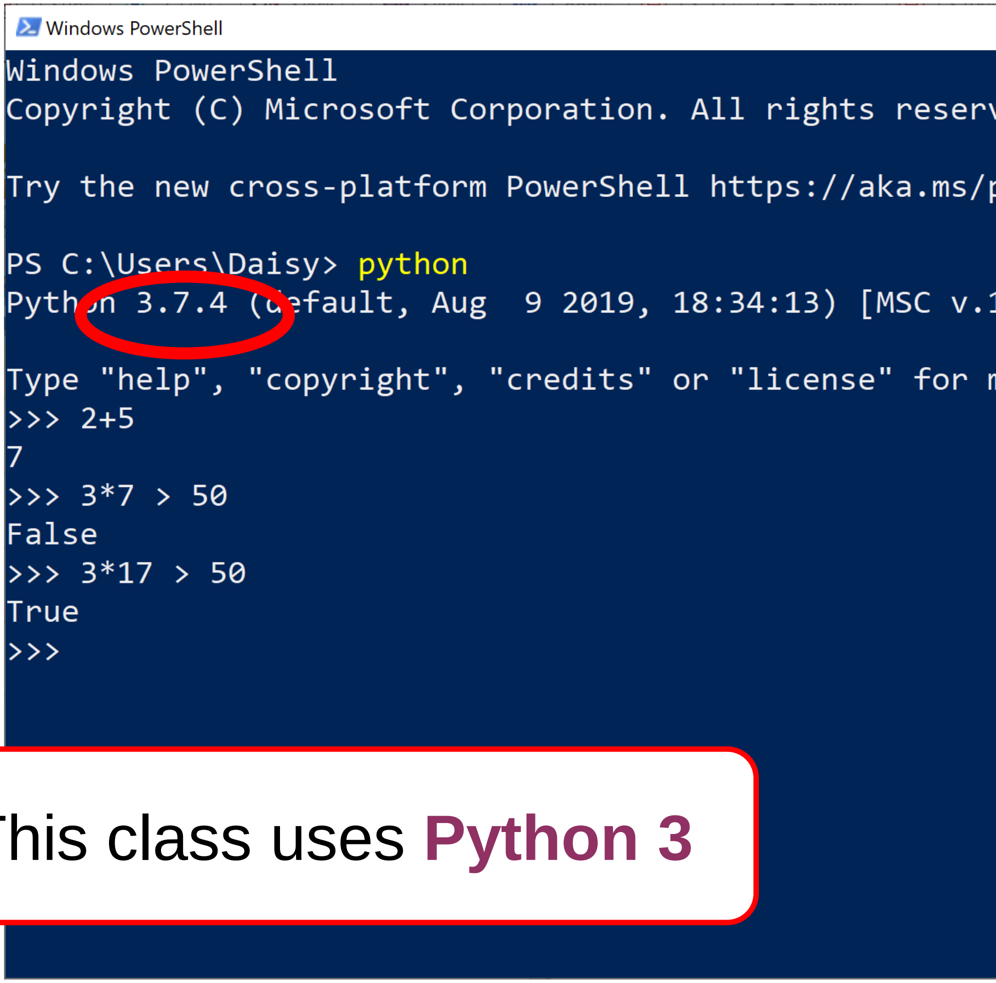
## Getting started with Python

- Designed to be used from the "command line"
  - OS X/Linux: **Terminal**
  - Windows: **PowerShell**

    (old: **Command Prompt**)
  - Purpose of the first lab
- Install, then type "python"
  - Starts the *interactive mode*
  - Type commands at **>>>**
- First experiments:

  evaluate *expressions*

```
>>> terminal time >>>
```


Windows PowerShell

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserv

Try the new cross-platform PowerShell https://aka.ms/p

PS C:\Users\Daisy> python
Python 3.7.4 (default, Aug  9 2019, 18:34:13) [MSC v.1

Type "help", "copyright", "credits" or "license" for m
>>> 2+5
7
>>> 3*7 > 50
False
>>> 3*17 > 50
True
>>>
```

This class uses **Python 3**

Python not installed yet? Use a python interactive shell at www.python.org/shell

## Storing and computing data

What data might we want to work with?

(What's on your computer?)

True

42

"apple"

🟪

False

$3.0 * 10^8$

"Tower Road"

🟪

0.00001

"awb93"

⬛

14850

## Expressions

An expression **represents** something

- Python ***evaluates it*** (turns it into a value)
- Similar to a calculator

Examples:

- 2.3

Literal
(evaluates to self)

- (3 * 7 + 2) * 0.1

An expression with four literals and some operators

## Types

A type is a set of values and the operations on these values

- Examples of operations: +, –, /, *
- Meaning of operations depends on type

**Memorize this definition!**

## How to tell the type of a value?

Command: **`type`**`(<value>)`

Example:

```
>>> type(2)
<class 'int'>
```

```
>>> terminal time >>>
```

## Type: **`float`** (floating point number)

**Values:** (approximations of) real numbers

- With a ".": a **float** literal (*e.g.*, 2.0)
- Without a decimal: an **int** literal (*e.g.*, 2)

to power of

**Operations:** +, –, *, /, **, unary –

***Note:*** operator meaning can change from type to type

**Exponent notation** useful for large (or small) values

- –22.51e6 is $-22.51 * 10^{6}$ or –22510000
- 22.51e–6 is $22.51 * 10^{-6}$ or 0.00002251

## Floating Point Errors

Python cannot store most real numbers exactly

- Similar to problem of writing 1/3 with decimals

Approximation results in **representation error**

- When combined in expressions, the error can get worse
- **Example**: `0.1 + 0.2`

```
>>> terminal time >>>
```

## Type: **`int`** (integers)

**Values:** …, –3, –2, –1, 0, 1, 2, 3, 4, 5, …

More Examples: 1, 45, 43028030

(no commas or periods)

division (technically a float operator)

"floor division": *divide then round down*

**Operations:** +, –, *, **, /, //, %, unary –

remainder

```
>>> terminal time >>>
```

## Type: **bool** (boolean)

### **Values:** `True`, `False`

- Boolean literals True and False (must be capitalized)

### **Operations: not, and, or**

- **not** b: **True** if b is false and **False** if b is true
- b **and** c: **True** if both b and c are true; **False** otherwise
- b **or** c: **True** if b is true or c is true; **False** otherwise

Often come from comparing **int** or **float** values

- Order comparison: k **<** j k **<=** j k **>=** j k **>** j
- Equality, inequality: k **==** j k **!=** j

"=" means something else!

## Class Materials

*sash means 2nd ed*

**Textbook.** *Think Python, **2nd ed.*** by Allen Downey

O'REILLY
2nd Edition
Updated for Python 3
Think Python
HOW TO THINK LIKE A COMPUTER SCIENTIST
Allen B. Downey

- *Supplemental;* does not replace lecture
- Available for free as PDF or eBook
- First edition is for the Python 2 (bad!)

**Python.** Necessary if using your own computer

- See course website for how to install

## Things to do before next class

1. Read textbook
   - Ch 1, Sections 2.1-2.3, 2.5, 2.6
2. Watch lesson videos
3. (If using your own computer) Install Python **following instructions on the website**
4. Attend lab on Tues/Wedn!

Lots of information on the website!

- Class announcements
- Consultant calendar
- Reading/Lessons schedule
- Lecture slides
- Exam dates
- Installation instructions

Read it thoroughly:

[www.cs.cornell.edu/courses/cs1110/2021sp/](www.cs.cornell.edu/courses/cs1110/2021sp/)

## Communication

cs1110-prof@cornell.edu

- Includes: both professors & head TA
- **For sensitive correspondence.** Don’t email one prof, or both separately.

cs1110-staff@cornell.edu

- Includes: both profs, admin assistant, graduate TAs, head consultants
- **For time sensitive correspondence (i.e., emergencies).** E.g., Nobody at office hours.

Ed Discussion: online forum (start from link on course website)

Email from us: please check your spam filters for mail from kdf4, LJL2, cs1110-prof, or with [CS1110] in the subject line