

October 5, 2019 Urbana, IL

IAHR & IWRA UIUC Python Workshop: Basic Python

## Before we begin...

- If you haven't already, go to <a href="https://github.com/sushobhansen/aci-python-workshop">https://github.com/sushobhansen/aci-python-workshop</a> for course materials (<a href="https://github.com/sushobhansen/aci-python-workshop">use the hydro-lab branch</a>)
- Download and install Anaconda: <a href="https://www.anaconda.com/distribution/">https://www.anaconda.com/distribution/</a>
- Or use an online IDE: <a href="https://repl.it/languages/python3">https://repl.it/languages/python3</a>

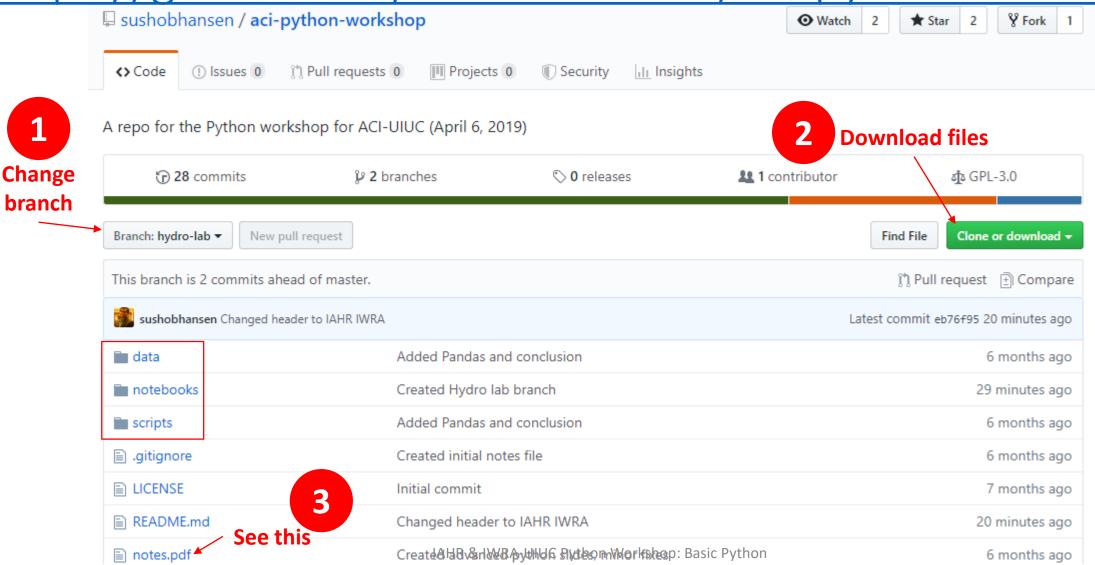
#### Who am 1?



- Sushobhan Sen
- Doctoral candidate in Civil Engineering (Transportation) – I study stratified urban boundary layer flows
- Languages I know: FORTARN, C/C++/C#, Python,
   Visual Basic, HTML, CSS, JavaScript, Latex, Bash
  - (I also speak a few human languages)
- More at: <a href="https://sushobhansen.github.io/">https://sushobhansen.github.io/</a>
- E-mail: sen6@illinois.edu

### Workshop Website:

https://github.com/sushobhansen/aci-python-workshop



## Learning Objectives: Basic Python

- At the end of this workshop, participants will be able to:
  - List the types of Python variables and define them
  - Add control statements and loops to their programs
  - Define and use functions
  - Define and use object oriented programming (if we have time)
- We'll also solve a problem using all these skills

# Why Python?

- According to IEEE, it's the most popular programming language today
- Simple, extensible, powerful
- Latest version: Python 3.7 (Python 3.x is not backwards compatible with Python 2.x, so stick to 3.x)

#### Variables

Data	Syntax	Description	Comments
Type			
Integer	x = 5	Signed integer	Use int to typecast,
			if valid
Float	x = 5.	IEEE floating point	Use float to type-
		number	cast, if valid
Complex	x =	Complex number	Use complex to type-
	3.1+4.6j		cast, if valid
String	x = 'Hello	Strings are always	Use str to typecast,
	World!'	enclosed within quo-	slicing operator valid
		tation marks	
List	x = [1,	Mutable list of vari-	Use list() to type-
	2.2,	ables of any type	cast, if valid
	'otter']		
Tuple	x = (1,	Immutable tuple of	Use tuple() to type-
	2.2,	variables of any tupe	cast, if valid
	'otter')		
Dictionary	x =	Key-value pairs	Use get() to get
	{'one':1,		value from key, use
	'two':2}		dict() to typecast,
			if valid
Bool	x = True	Boolean value	Python uses key-
			words True and
			False, not 1 and 0

#### Variable Names

- 1. The name **cannot** contain spaces consider using an underscore character or camelCase instead for readability
- 2. The name cannot start with a number, but can contain a number anywhere
- 3. The name is case-sensitive, so temp and Temp are different variables
- 4. Python has a number of reserved keywords (see documentation, of just follow along and you'll get the hang of it), which cannot be used as names



# Mutability

- Immutable: Most objects (int, float, complex, str, bool, tuple)
- Mutable: Lists and dictionaries



## Operators

- Arithmetic: +, -, \*, /, %, \*\*, //
- Comparison: >, <, ==, !=, >=, <=
- Logical: and, or, not
- Bitwise: &, |, ~, ^, <<, >>
- **Assignment:** =, any combination of arithmetic or bitwise with =
- Special: is, is not, in, not in



#### **Control Statements**

- 1. Conditional statements: Better known as if-else blocks, these test a condition before executing a line
- 2. **Loops**: These repeatedly execute a line for a certain number of times or till a break condition is met

# A simple if statement

```
1 x = 6
2 if x<5:
3    print('x is less than 5')
4    print('Finished evaluating if block')
5 print('Finished this code block')</pre>
```

Notice the colon and indentation: very important!



## Loops

- While loop: Repeat till a condition is met
- For loop: Repeat over a fixed number of iterations

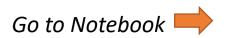
## For loops

- Loop over an iterator
  - An iterator is an object that contains a fixed number of values (any values)
  - Iterate through the values in the iterator, not just a fixed number of times (unlike C++)

```
1 x = range(3)
2 print(list(x))
3 for i in x:
4 print('The value of i is ',i)
```

## Break, continue, pass

- Break: When executed, break out of loop entirely without any more iterations
- Continue: When executed, skip everything after that line and then proceed to next iteration
- Pass: When executed, continue to next line (as if nothing happened)



#### **Functions**

• Functions are bits of code that are separated from the main function (they have their own scope), but can be called at any point by the main function to perform a specific task.

```
function def statement
def statement
k = 'Hello'
print_input(k)

function def statement
k = 'Hello'
print_input(k)
```

# Example: System of Linear Equations

• Solve: 
$$\begin{cases} ax + by = e \\ cx + dy = f \end{cases} \Rightarrow \begin{cases} x = (de - bf)/(ad - bc) \\ y = (af - ce)/(ad - bc) \end{cases}$$

• Unique solution exists only if  $ad - bc \neq 0$ 

## Default Args

# Classes and Objects

- Class: Set of related variables and functions that are grouped together
- Object: An instance of a class

```
class Person:
                          def __init__(self,name,age,height_cm): ← Constructor
                   Dot
                               self.name = name
                               self.age = age
                operator
                               self.height_cm = height_cm #in centimeters
                          def print_name(self):
                               print(self.name)
Defining a
                          def print_age(self):
                   10
  class
                               print(self.age)
                   11
                                                      Always pass self as first arg
                   12
                   13
                          def print_height_cm(self):
                   14
                               print(self.height/cm)
                   15
                   16
                          def height_ftin(self):
                   17
                               inches = self.height_cm/2.54
                   18
                               feet = inches//12
                   19
                               inches = inches%12
                   20
                               return feet, inches
```

#### Inheritance

 A class can inherit all the functions and variables from a base class, and then add or modify them

#### **Person class** class Student(Person): def \_\_init\_\_(self,name,age,height\_cm,GPA,level): Person.\_\_init\_\_(self,name,age,height\_cm) self.GPA = GPAself.level = level def print\_name(self,status='regular'): print(self.name,' STUDENT: ',status) 9 **Polymorphism** def print\_GPA(self): **ABCs** print(self.GPA) def print\_level(self): print(self.level) 15

Student class inherits from

## Challenge

- Use Python to write a prime\_check() function, which takes in an integer and returns the following:
  - Negative integers or zero: "Invalid input"
  - If 1 or 2: "Prime number"
  - Any other integer: "This is a prime number" or "This is not a prime number"
- Algorithm for prime numbers:
  - Sequentially divide the integer n by integers between 2 and n/2
  - If at any point, the remainder is zero, it is a prime number, otherwise it isn't

#### Further Resources

- Python Documentation: <a href="https://docs.python.org/3/">https://docs.python.org/3/</a>
- Free Python course on EdX: <a href="https://www.edx.org/professional-certificate/introduction-to-computing-in-python">https://www.edx.org/professional-certificate/introduction-to-computing-in-python</a>
- Free e-Book: https://www.cs.uky.edu/~keen/115/Haltermanpythonbook.pdf
- Stick around the Advanced Python session in the afternoon