W200 - Week 2

Starting out with Python

Agenda

1. Last-week checkin

2. Review some main points: Running Python.

   1. Expressions, Objects, Variables

3. Activities:

   1. Strings, Control-of-Flow (if, while)

   2. Breakout room

Berkeley
SCHOOL OF INFORMATION

W200-Python
Fall 2018

Check-In

- How did it go setting up your environment? Works for you?
The TAs check and feedback weekly assignments; instructors review, grade, feedback projects and exam. Assignments are posted on GitHub by the TAs on Monday/Tuesday evening.

- Python 3.x (3.6, 3.7) [no PyCharm or Spyder]

- GitHub [Your repo ready?]

- Bash [http://hyperpolyglot.org/unix-shells]

- Jupyter

MIDS-INFO-W18

Repositories 484   People 420   Teams

Search repositories...

github-playground Private
Python ★ 1 ⑂ 16 Updated an hour ago

LauraPintosREPO Private
Updated an hour ago

Found or created something cool?
Share it in the Playground

ErrettHobbsREPO Private
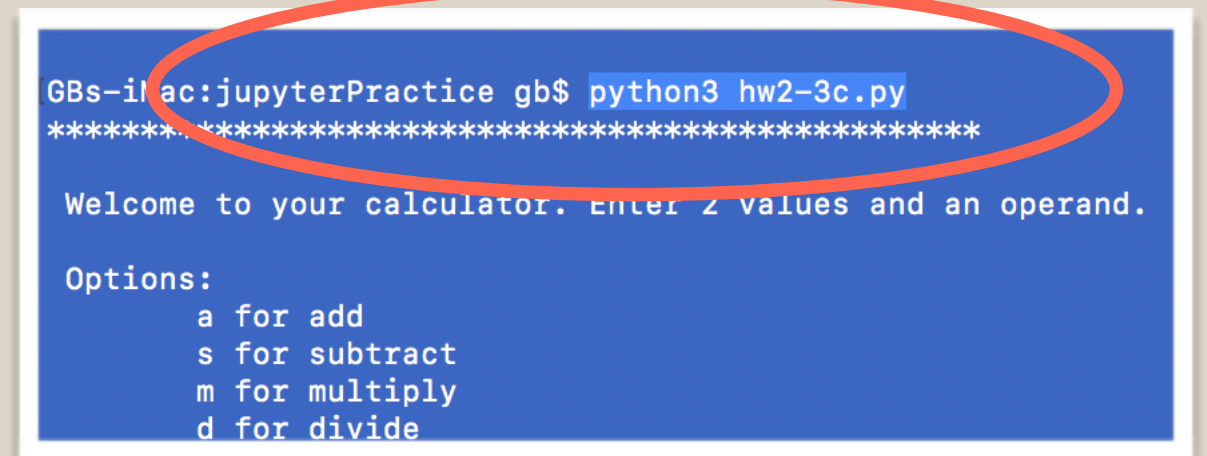Shell Updated 3 hours ago

Syllabus & Schedule

Python for Data Science: Fall 2018						All due dates are tentative and may be changed by instructors. Homework due dates are 11:59pm PST the night before live session.					
Mon	**Tues**	**Weds**	**Thurs**	**Async Unit**	**Sync Week**	**Async to Review (Prior to Class)**	**Projects (20% each)**	**Exams (10% each)**	**HW Assigned (30% total)**	**HW Due**	**Notes**
Sep 3	Sep 4	Sep 5	Sep 6	1	1	Introduction to Programming, the Command Line, and Source Control			unit 1		A make-up class will be scheduled for Monday class.*
Sep 6											[This is the make-up for Monday 4 pm session.]
Sep 10	Sep 11	Sep 12	Sep 13	2	2	Starting Out with Python			unit 2	unit 1	
Sep 17	Sep 18	Sep 19	Sep 20	3	3	Sequence Types and Dictionaries			unit 3	unit 2	9/17/2018 - Last day to add or drop a class
Sep 24	Sep 25	Sep 26	Sep 27	4	4	More About Control and Algorithms			unit 4	unit 3	
Oct 1	Oct 2	Oct 3	Oct 4	5	5	Functions			unit 5	unit 4	
Oct 8	Oct 9	Oct 10	Oct 11	6	6	Complexity	Project 1 Assigned		scrabble	unit 5	
Oct 15	Oct 16	Oct 17	Oct 18	7	7	Classes			unit 7	scrabble	
Oct 22	Oct 23	Oct 24	Oct 25	8	8	Object-Oriented Programming	Project 1 Final Proposal Due	Exam 1 Start	x	unit 7	
Oct 29	Oct 30	Oct 31	Nov 1	9	9	Working With Text and Binary Data		Exam 1 Due	x		
Nov 5 - 9 Fall Break & Immersion											
Nov 12	Nov 13	Nov 14	Nov 15	10	10	NumPy	Project 1 Presentations		unit 9 / HW10		A make-up class will be scheduled for Monday Class
Nov 19	Nov 20	Nov 21	**Nov 22**	11	11	Data Analysis With Pandas	Project 2 Assigned		unit 10 / HW11	unit 9 / HW10	A make-up class will be scheduled for the Thursday Classes
Nov 26	Nov 27	Nov 28	Nov 29	12	12	Plotting and Visualization	Project 2 Proposal Due		unit 11 / HW12	unit 10 / HW11	
Dec 3	Dec 4	Dec 5	Dec 6	13	13	Pandas Aggregation and Group Operations		Exam 2 Start	x	unit 11 / HW12	
Dec 10	Dec 11	Dec 12	Dec 13	14	14	Testing	Project 2 Presentations!	Exam 2 Due	x		Last Day of Class. bring beer Congratulations!
Last Day of Instruction - December 14											

Live Schedule: https://docs.google.com/spreadsheets/d/1sVV7-4OHZ-EDNqkMJ55OPUfz_QLJ4LZNuZl-cRaxgV0/edit?usp=sharing

Syllabus: https://docs.google.com/document/d/1_lLP7iM11IWNdtZL80-axCznLZUGz4oaVxB4FuryCH0/edit?usp=sharing

Running python

- Command line

  - Use a text editor (Atom, NotePad, TextWrangler, etc.) and save as a .py file.

  - Jupyter Notebook


```
GBs-iMac:jupyterPractice gb$ python3 hw2-3c.py
*********************************************************

 Welcome to your calculator. Enter 2 values and an operand.

 Options:
        a for add
        s for subtract
        m for multiply
        d for divide
```

Berkeley
SCHOOL OF INFORMATION

Python expressions

Assignment	**=**	`x = 5`
Addition	+	`x + 5`
Subtraction	–	`x – 5`
Division	/	`x / 5`
Multiply	*	`x * 5`
Exponent	**	`x ** 5`
Cumulative	+=	`x += 5` `x = x + 5`

|		
Equivalency	**==**	`x == 5`
Cumulative	–=	`x –= 2`
Integer division	//	`x // 5`
Modulus	%	`x % 2`

Python & Objects

- Everything in python is an object … and every object has a type (class) ["object space"]. These details will be covered in more detail in later parts of the course, so if it is confusing we'll circle back.
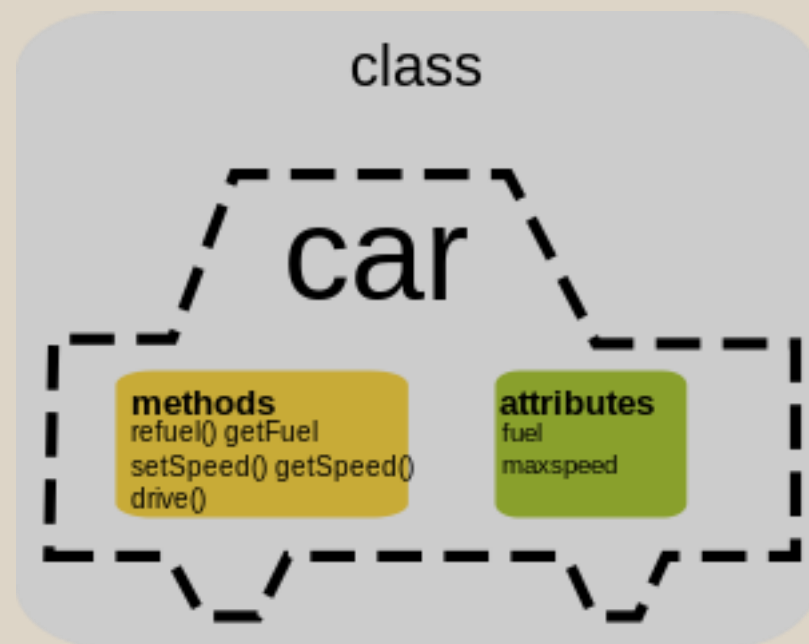
"Real world" things expressed in a computer.
The properties/facets of the object are attributes; what the class can do (like verbs) are the methods.
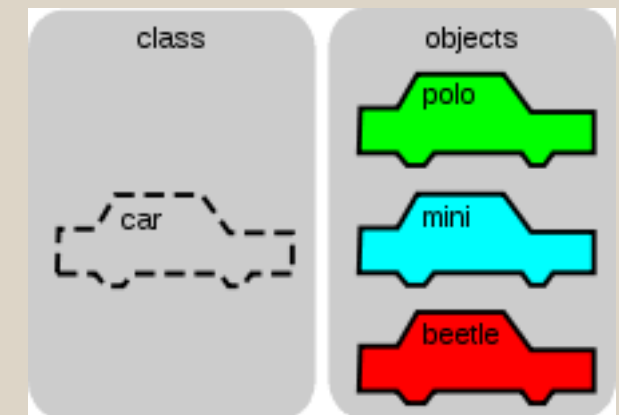
Copy the class ("constructors") and tailor as you need to. Encapsulation, Polymorphism, Inheritance
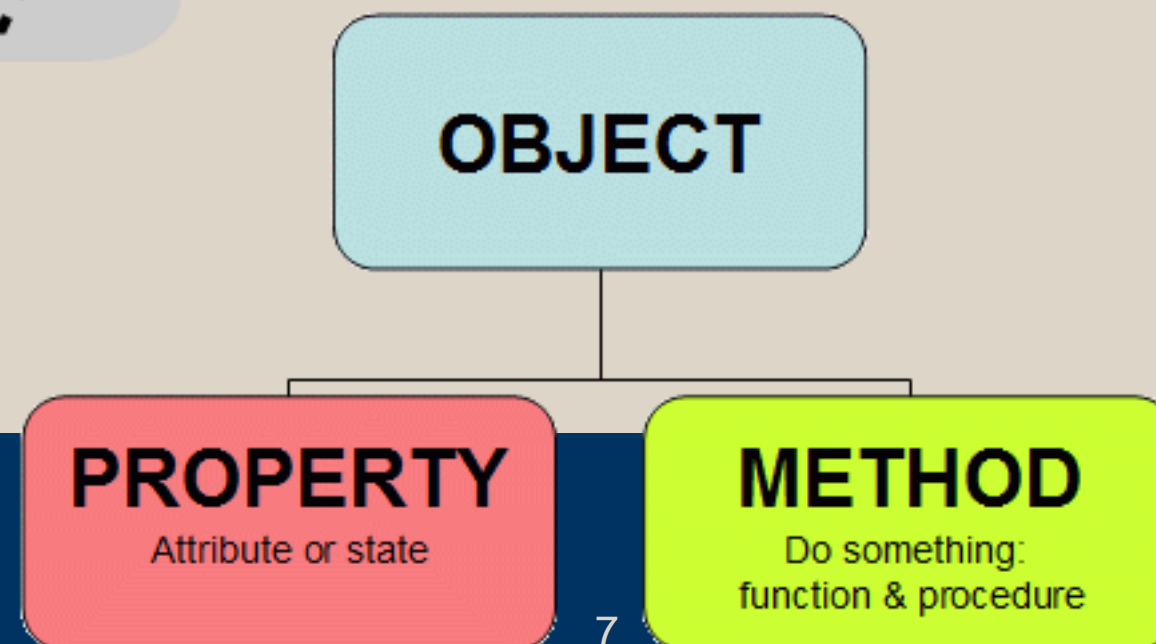
Python Objects & Types

- `x = 5` ⟶ `type(x)` ⟶ `int`

- Boolean (`True, False`; 0 1) bool

```
>>>
>>> x = 5
>>> type(x)
<class 'int'>
>>>
```

- Integers (**`int`**) Counting numbers; 1, 2, 3, ... n

- Floating point numbers (**`float`**; e.g., 3.14159)

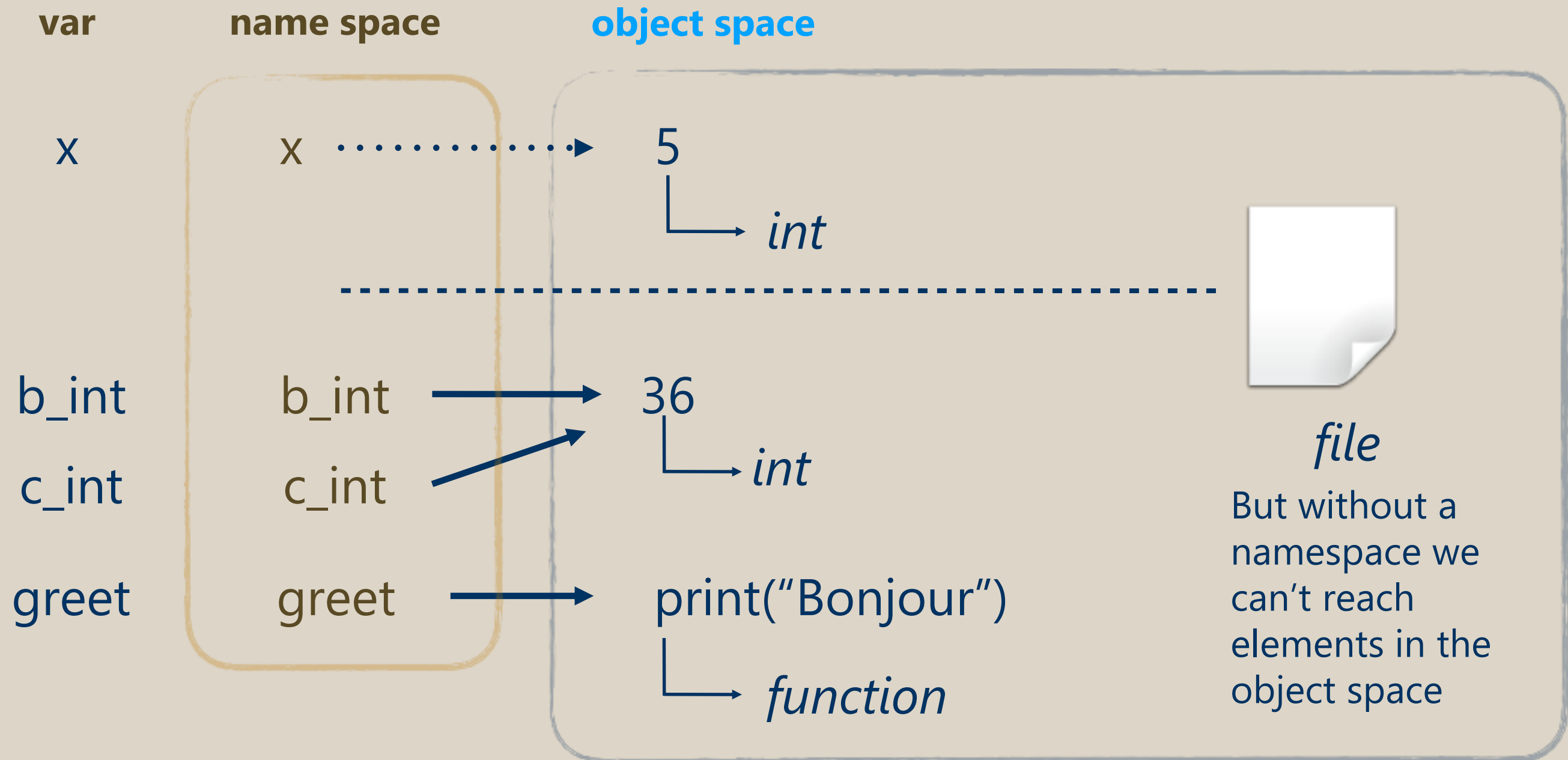- String (**`str`**) - A sequence of characters.
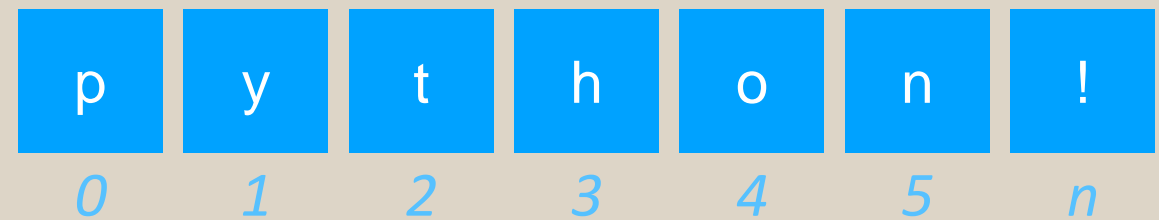
Type cast: convert data from one type to another, e.g.,
- `n = int(input("Enter a integer: "))`

Berkeley
SCHOOL OF INFORMATION

var | name space | object space

var | **name space** | **object space**

Variables go into a special object called a "name space." Think of "scope & visibility" of vars and how they're accessed in the code.

x | x ....▸ 5
  | |_____▸ *int*

b_int | b_int ──────▸ 36
c_int | c_int ──────▸ |_____▸ *int*

file

greet | greet ──────▸ print("Bonjour")
  | |_____▸ *function*

But without a namespace we can't reach elements in the object space

Berkeley
SCHOOL OF INFORMATION

String Objects

- A string is a "sequence object" [imagine the memory address of this string is 0 and is n bytes long. - Locate entire strings, individual letters, range of letters]

- Strings can be joined (or "concatenated") and manipulated:
  1. concatenation: "cat" + "dog" outputs "catdog"
  2. multiplication of chars: "-"*10   outputs   ——————
  3. `<string>.upper()`, e.g., `x = "cat"     print(x.upper())`
  4. `<string>.lower()`
  5. `s = input("Type here:")`
  6. `s_cast = int(input("Type here (integers only):"))`

Strings – Special Chars

- Single and double quotes:

```
' '             # single quotes (the dagger kind)
"   "           # double quotes (notice the pairing)

t = input("Welcome, you're here! ")
                # double quotes surrounding string
                # so the single quote is safe
s = input('Welcome, you\'re here!')
                # single quotes, so middle one needs
                # to be "escaped"
```

- Block quotes   """

- \ escape sequence … \n (newline) \t (tab) \'

- print("here is a string.", end = '')

Berkeley
SCHOOL OF INFORMATION

String Slicing: using indices

p	y	t	h	o	n	!

	0	1	2	3	4	5	n	
`[0]`					from the start of the string			
`[-1]`					one letter from the end of the string			
`[0:3]`					from the start of the string to the 4th letter			
`[1:-1]`					from 1 to the next to last letter			
`[1:5:2]`					from 1 to 5, by 2 *at a time*			
`[:-1]`					beginning (:) to second to last letter (*n* - 1)			
`[:]`					entire string			
`[::-1]`					entire string, reversed			

Berkeley
SCHOOL OF INFORMATION

W200-Python
Fall 2018

Breakout Session (10-12 min)

1. Fire up what you downloaded for week 2 in Jupyter and create a string variable that prints exactly

```
The "trouble with
Tribbles" is that they
\\\EAT/// too many MREs.
```

2. Using one line of Python code, slide your variable from part 1 to print "selbbirT" 300 times.

```
selbbirT selbbirT selbbirT …
```

Control of Flow (if, while)

If statements

Nested if

while loop

Nested loops

```
if xxx:
    doX()
elif yyy:
    doY()
else:
    doZ()
```

```
if x > 2:
    print("x is greater than 2")
elif x < 0:
    print("x is negative")
else:
    print("x is less than 2 but still positive")
```

```
countdown = 5
while countdown > 0:
    print(countdown)
    countdown -= 1
print("Blast off!")
```

We'll see several other techniques, such as for x in range() when we work with lists and dictionaries.

Nested Loop | Repeating an action

```
row = int(input("Enter an integer: "))

# while row >= 0:

j = 0
while j <= row:
    print(j, end=" ")
    j += 1
```

Enter an integer: 5
0 1 2 3 4 5

Enter an integer: 5
0 1 2 3 4 5
0 1 2 3 4
0 1 2 3
0 1 2
0 1
0

```
row = int(input("Enter an integer: "))

while row >= 0:
    # inner loop
    j = 0
    while j <= row:
        print(j, end=" ")
        j += 1
    print("")
    row -= 1
```

Berkeley
SCHOOL OF INFORMATION

15

Breakout room

```
ans = input('do you have 8 legs?')
if ans == "yes":
    print("you are a spider")
else:
    ans = input("do you have 4 legs?")
    if ans == "yes":
        print('you are a quad')
    else:
        print("you are a bicycle.")
```

Breakout rooms

- Make a calculator

- Work with conditionals (if statements)

- Save a .py file and execute it.

```
Enter a first number : 4
Enter a second number: 3
Enter operand: "a", "s", "d" or "m": a
4.0 + 3.0  =  7.0
```

```
Enter a menu option: [add, sub, div, multi]:     add
Enter the first value:  3
Enter the second value: 2

3.0 + 2.0 = 5.0

Great job 👍
```

That's it! Enjoy a great week and keep up
with your drills and assignments.