# Python Basics: Part 2

Benjamin Rose

May 31, 2018

## Learning Goals

1. Get files via `git`.
2. Be able to run and understand basic code
3. Be able to explain what is and how *control flow* works in python
4. Understand the similarities and differences in the basic python collection types
5. Learn 10 more Python terms.

## Git and GitHub

- Get a Github account
- Go to Github classroom link, from website
- GitKraken
- Log in and clone your new repo

# Review

`stack_example.py`

1. What does this do?
2. How can you improve it?

# `bisection.py`

Run it with

```
$ python3 bisection.py
```

Make sure you are in the correct directory!

# Review Data types

## Booleans

`True` or `False` within python

- `True` can also mean non-zero or non-empty or has-length (don't worry too
  much about this now)
- can use `and`, `or`, `not`, and group comparisons logically
- Unlikely to hard code these into the software

## Comparison Operators

- These operators evaluate to `True` or `False`

1. `==` `!=`
2. `>` `<`
3. `>=` `<=`

# Controlling program flow

*pseudo-code*

```
if condition is True:
    print('true')
elif condition is False:
    print('false')
else:
    print('you will never get here')
```

# Control Flow Example

```python
x = 5
if x > 10:
    print('x is large')
elif x > 2:
    print('x is medium')
elif x >= 0:
    print('x is small')
else:  # x < 0
    print('x is negative')
```

## Data Structures

lists: `l = [1, 2, 3]`

tuples: `t = (1, 2, 3)`

sets: `s = set([1, 2, 3])`

dictionary: `d = {'a': 1, 'b': 2, 'c': 3}`

## Reading for next Tuesday

We have a few more basics before we are able to start writing useful programs.

Think Python: Ch 7, 8.3, and 14.1-.4

*note:* this week we covered Chs. 2, 3 and 5