Python 2.0 vs Python 3.0: Different versions of python, but Python 2.0 is used more frequently

Python is Multi-Paradigm:

* Object Oriented
* Functional
* Imperative

Python is Opinionated:

* Clear style guide: PEP-8
* The Zen of Python: *$python –c ‘import this’*
  + Pythonic Code
  + The better or “right way” to write python

Python is Dynamically Typed (But Can Still Be Typed)

* Does not have defined data types
* You can find the type of the variable:
  + x = 123
  + type(x)
  + <type ‘int’>
* Variables checked at run-time, not compilation-type
* Interpreted language
* “Duck-typing” – can act in a certain way

*Statically Typed:* Declared in advanced

*Dynamically Typed:* Not needed to be declared

Strong:

Weak:

Python is Clean, but Whitespace Matters

* No line-ending semicolons, but one line at a time
* No mess of nested braces {{}}, but spacing matters

Python is Fully-Loaded

* Standard libraries: json, csv, re, math, datetime, logging, random…
* Large community
* Organized into modules
  + import X 🡪 import math
  + import Y from X 🡪 import sin() from math

Why not Python

* Not the fastest language (a little slower)
* Uses more memory
* Not “new” or “trendy”

Anaconda:

* Separate from any other Python installation
  + iPython shell
  + iPython notebook server
  + Spyder IDE
* Popular in data science
* Separate package system from “normal” Python
* conda install vs pip install

REPL: **R**ead **E**valuate **P**rint **L**oop 🡪 Makes debugging easier

* Strings: my\_list = [1,2,3,4,5]
  + var [ starting index : ending index : stop (optional) ]
  + num\_list[::2]
  + [1, 3, 5]
* Dictionaries: mydicitonary = {'a':'Alexandra', 'b':'Bertrand'}
  + my\_dictionary['a']
  + 'Alexandra'
* IF-ELSE
  + >>> if user\_input==password:
  + ... print("You got it!")
  + ... else:
  + ... print("Not allowed in!")
  + ...
  + Not allowed in!
* Boolean
  + >>> 5>3
  + True
* elif: else if statement
* JSON stores data similar to Python
* for <num> in <numlist>:
  + >>> for num in num\_list:
  + ... print(num)
  + ...
  + 1
  + 2
  + 3
  + 4
  + 5
* >>> square\_list = []
* None is NULL in python
* List comprehension:
  + >>> [x\*x\*x for x in num\_list]
  + [1, 8, 27, 64, 125]
* Import the math dictionary:
  + >>> import math
  + >>> math.sin(5)
  + -0.9589242746631385
* >>> from math import sin
* You can overwrite the function (so best practice is to use math.sin()
* Pretty Print: pprint

Code School: Free Python Course

Codecademy: Free Python Course

Data Science: NumPy, Pandas, Scikit-learn, Statsmodels, NLTK, Seaborn, Matplotlib

Web Development: Flask, Django, Pyramid, Requests, SQLAlchemy, Jinja2