5
Python Basics

History of Python

- Invented by Guido van Rossum in late 1980's and released in 1991. Guido is the benevolent dictator for life for Python.
- The Python language was named after *Monty Python's Flying Circus*, not the snake
- There are 2 major versions of Python: 2.7 and 3.6
- In this class we will use Python 2.7
- Python is often used in mathematical and scientific data based applications
- Python is used by Google, Wikipedia, NASA, and Reddit, among others

Guido van Rossum

- Developed Python in an effort to improve upon a language called ABC
- Used to work for Google, works at Dropbox as of January 2017

External libraries

- PyPI - the Python Package Index - consolidated repository of 3rd party software libraries, equivalent to Perl's CPAN - https://pypi.python.org/pypi
- Django - Popular Python-based web framework - https://www.djangoproject.com/
- Other popular Python libraries include
  - NumPy - http://www.numpy.org/
  - PyQt - https://riverbankcomputing.com/software/pyqt/intro
  - PyGTK - http://www.pygtk.org/
  - Twisted - https://github.com/twisted/twisted
  - PyGame - http://www.pygame.org/news

Python Philosophy and Approach

- Python is intended to be a highly readable language.
- Python prioritizes simplicity and readability over speed.
- From Wikipedia - Python's philosophy rejects the Perl "there is more than one way to do it" approach to language design in favor of "there should be one—and preferably only one—obvious way to do it".
- Compared to Perl, Python syntax is uncluttered.
- The Zen of Python - https://www.python.org/dev/peps/pep-0020/
- The philosophy of Python can be summarized as “Simple is better”.

Python language basics

- No curly braces or semicolons, but rather indentation is used for blocks
- if...elif...else (a contraction of else-if)
- try...except...finally
- def - defines a function or method
- class - attaches a block of code to a named class
- No auto-increment i++ or auto-decrement i--. You have to use i += 1 or i -= 1
- Python’s reference implementation is called CPython.

Python language basics

- The == operator compares objects by value (do they have the same value?).
- The “is” operator compares objects by reference (do they point to the same memory location?).
- Uses ‘and’, ‘or’, ‘not’ for boolean logic, instead of symbolic operators &&, ||, !
- Python uses tuples, lists, and dictionaries as container objects.
- No ternary operator ?: You have to use if-else for that instead

```
$x = 4;
print $x < 0 ? ‘negative’ : ‘non-negative’, “\n”;
```

Other Python idioms

- List comprehensions
- Duck typing - If it walks like a duck, quacks like a duck, it’s probably a duck.
- Classes, methods, and the self keyword

Python variables compared to Perl variables

- In Python variables are assigned a scalar value that holds a reference to (or can think of “pointing to”) some other data object.

- Python references are similar to references in Java/C#, or pointers in C/C++.

- Every data object in Python has the following 3 properties:
  - An Identity - retrieved by a built-in function called id()
  - A Type - retrieved by a built-in function called type()
  - A Value - automatically retrieved by any expression that evaluates the object

Numbers

- Plain Integer - Same max size as the underlying C library signed long
- Long Integer - Unlimited size (machine/OS limit)
  - Integers in Python always stay integers regardless of size they are never ‘promoted’ to floats
- Integer division - result is “floored” in Python 2.x, but in 3.0 integer division results in true division (i.e. ½ will result in 0.5)
- The // operator always returns the floor of any combination of operand number types
- Floating-point number - double-precision as understood by the C library, usually most machines will allocate 8 bytes for this
- Complex number - The imaginary part is followed by a j or J.

Immutability

- Python’s container types are String, Unicode string, Tuple, Xrange, List, and Dictionaries
- Only Lists and Dictionaries are mutable, all other types are immutable
- Immutable means that once the object is created, it never changes until it’s destroyed.
- Any operations on immutable objects that appear to be making changes are really making a complete copy of the object and returning that new object.
- Mutable objects can be changed “in-place”, meaning they are truly changed, not copied.

Shallow vs. Deep copying

- Shallow Copy - A new reference is created. The underlying object is the same object, only now another named reference “points-to” that same original object’s location in memory. Any updates using either the old or new reference will change the same object.

- Deep Copy - The whole object is copied into a new location in memory. The new reference “points-to” the new object’s location in memory. Any updates using the new reference will NOT impact the original object or its reference. Likewise, any changes to the original object will not impact the new reference or the object it references.

Strings

- Python strings can be single or double-quoted, they both work the same
- Python also has triple-quoted strings for when you want to span multiple lines
- Raw strings - begin with “r” and ignore escape sequences like @ strings in C#
- There is no string interpolation in Python like in Perl, although Python has format strings that can do basically the same thing.

```
x = “perl”
y = “python
print “%s is not %s” % (x, y)          # perl is not python
```

Tuples

- A tuple is sequence of objects separated by the comma operator.
- Tuples can be defined with or without parentheses.
- Tuples may contain objects of different types.
- Tuples use zero-based indexing like arrays.
- A singleton tuple must have a trailer comma: x = (1,)

Lists

- As with tuples, lists use zero-based indexing.
- Lists are declared with enclosing square brackets and commas separating values.
- The replication operator can be used to define multiple recurrences of the same value.
- An entire copy of the list can be returned using the syntax s[:] where s is a list.
- Lists can be sorted in-place, since they are mutable objects.

Xrange Sequences

- Generates a sequence based on a pattern in the constructor: Ex: xrange(0, 8, 2) will generate the sequence 0, 2, 4, 6
- The above xrange does not generate 0, 2, 4, 6, 8 because it stops before reaching the 2nd argument.
- Xrange sequences are very memory efficient because they generate their numbers on demand, or as-needed. So they don’t need to store the entire range of numbers in memory all at once.

Dictionaries

- Dictionaries in Python are equivalent to hashes in Perl.
- All keys must be immutable objects. However the dictionary object itself is mutable.
- Keys are not restricted to being strings like they are in Perl hashes.
- The order in which keys are stored is dependent on the underlying hash function that is used to translate the key into the stored value.