Python for Beginners

Getting started…
Values and types
Basic operators

Additional resources

- O'reilly books
  - *Learning Python*
  - *Bioinformatics Programming in Python*
  - *Python in a Nutshell*
  - *Programming Python*
  - *Python Cookbook*
  - *Fluent Python*
- Python.org
  - Downloads
  - In-depth documentation
- Tons of online tutorials

Obtaining Python and IDEs

- Python can be downloaded at python.org
  - Unix-based OSes (should) come with python
  - Have to install python on Windows systems
  - Check version of what you download
- Numerous IDEs available
  - PyCharm, Enthought/Canopy, Eclipse…
    - Available for all OSes and free (I think)

A few additional words

- Python versions
  - Python 3.x is available, but we'll learn python 2.x
  - No backwards compatibility
  - Most scientific software written in 2.x
- Integrated developer environments (IDEs)
  - Can write code in notepad, vi, gedit, etc…
  - IDEs provide a centralized location to write, edit, run, and debug code
  - Code completion and insight
  - Good for complex projects, but have a learning curve

Why Python?

- Powerful
- Portable
  - Programs can be run written and run under any OS
- Easy to…
  - Obtain/install
  - Read
  - Learn and use
- Lots of scientific software written in Python
  - For the above reasons

How to run a Python program/script

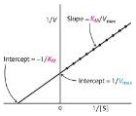
- Python is (sort of) an interpreted language
  - Programs are given to another program, the interpreter, to run them
  - Interpreter is usually called *python*
- Three main ways to run a python script:
  - Interactively (call *python* and then issue commands)
  - As a module (*python my_file.py)*
  - As an executable shell script
    - Linux based OSes only
    - #!/usr/bin/python

Michaelis-Menten kinetics

- Describes kinetics of many enzymes

$$v = \frac{v_{max}[S]}{[S] + K_m}$$

- Measure enzymatic activity (v) at varying substrate concentrations ([S]) to get V_{max} and K_m
- Can be obtained from Lineweaver-Burk plot



- Say we measure two perfect data points:
  - At [S] = 0.75, v = 57.9
  - At [S] = 2.10, v = 101.3
- What's K_m?

What is a program?

- Programs do things with stuff
- Stuff is usually some form of data
- Each piece of data has a value
- Each value has a type
- Values can be assigned to variables
- Take a simple python statement: a = 5
  - The variable a is assigned a value of 5
  - Both 5 and a are of type integer

Python types – Numeric

- Integers (int)
  - 42, 0, -165, etc
- Floating-point numbers (floats)
  - 1.23, 42.0, 6.023e23, 1.38E-23, etc
- Less used types…
  - Long
    - Numbers outside of the range of integers, 99999999999999999999L
  - Complex
    - Numbers with imaginary component, 3+4j

Basic numeric operators

- Addition: a + b 3 + 5 = 8
- Subtraction: a – b 5.0 – 3.0 = 2.0
- Multiplication: a * b 5 X 3 = 15
- Division: a / b 15 / 5 = 3
- Floor Division: a // b 16.0 // 5.0 = 3.0
- Modulo: a % b 16 % 5 = 1
- Exponentiation: a ** b 5 ** 3 = 125

- A combination of value(s) and operator(s) is an expression

Type promotion

- Types in python are dynamic
  - Variables can change type on the fly - promotion
- Taking the numeric operators as an example…
  - *Int* op *int* = *int*: 3 + 5 = 8
  - *Int* op *float* = *float*: 3 + 5.0 = 8.0 (the *int* gets promoted)
  - *Float* op *float* = *float*: 3.0 + 5.0 = 8.0
- Follow coercion rules
  - Very long list…

Exercises

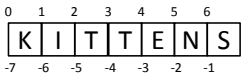
- What are the output values and types for these operations?
  - 2 / 5
  - 2 / 5.0
  - 16 % 5.0
  - 64 ** 0.5
  - 64 ** (1 / 2)
  - (1 / 2) + (1 / 2)
  - (1.0 / 2) + (1 / 2.0)

Python types - String

- String – ordered collection of characters
  - "Hello!", 'Hello', "y'all", 'abcd1234', ""
- Basic string operators
  - Concatenation: a + b "Hello " + "y'all" = "Hello y'all"
  - Repetition: a * b "Alright " * 3 = "Alright Alright Alright"
  - Subscription: a[i] "Alright"[0] = "A"
  - Slicing: a[i:j] "Alright"[0:3] = "Alr"

Strings – subscription/indexing

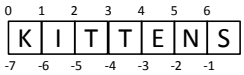
- Subscription/indexing retrieves an individual item from a sequence
  - Syntax: S[i]; S is the sequence, i is the index, S[i] is an element
  - Index/offset starts at 0
  - Or count backwards from -1
  - S = "KITTENS"



- S[0] = S[-7] = "K"
- S[2] = S[-5] = "T"
- S[4] = S[-3] = "E"
- S[6] = S[-1] = "S"

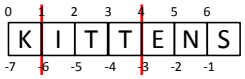
Strings - slicing

- Slicing retrieves a range of items from a sequence
  - Syntax: S[i:j]; i and j denote boundaries of slice
  - i defaults to 0 if not included; j defaults to length of the string
  - Extracts everything from the lower bound up to, but not including, the upper bound
  - Imagine slicing a loaf of bread…
  - S = "KITTENS"

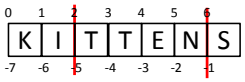


Strings - slicing

- i and j describe where the knife cuts the loaf…
- S[1:4] = “ITT”

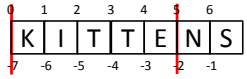


- S[2:-1] = “TTEN”

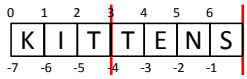


Strings - slicing

- S[:5] = “KITTE” (Left bound omitted, defaults to 0)



- S[3:] = “TENS” (Right bound omitted, defaults to end)



- S[:] = “KITTENS” (Both bounds omitted)

Exercises

- What is the output for these operations?
- S = “STRINGS”
  - S[5]
  - S[1:4]
  - S[:4]
  - S + 3
  - S + ‘3’

Python types - List

- List – ordered collection of anything
  - [1, 2, 3], [1, “two”, 3, 4.0], [1, [2, 3], ‘four’], []
  - Variable length, heterogenous, and nestable
- Similar to strings…
  - Addition: [1] + [2] = [1, 2]
  - Repetition: [1, 2] * 3 = [1, 2, 1, 2, 1, 2]
  - Subscription: [1, 2, 3][1] = 2
  - Slicing: [1, 2, 3, 4][1:3] = [2, 3]
- But also a little different…
  - Lists are mutable – they can be changed “in place”
  - Numbers and strings are immutable

Lists and mutability

- Strings are immutable
  - S = “KITTENS”
  - S[0] = “M” leads to an error
  - S = “M” + S[1:] works (S is now “MITTENS”)
- Lists are mutable
  - L = [1, 2, 3, 4]
  - L[0] = ‘ONE’ works (L is now [‘ONE’, 2, 3, 4])
  - L[1:3] = [“two”, “three”] works (L is now [1, ‘two’, ‘three’, 4]
  - L[1:3] = [] works (L is now [1, 4])

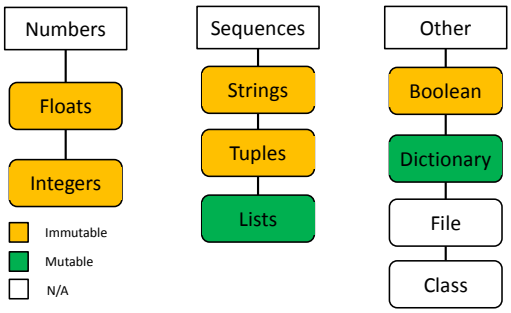
Python types – Tuple

- Tuples are immutable lists
  - (1, 2, 3), (1, “two”, 3, 4.0), (1, (2,3), four), ()
  - Follow all the rules of lists but can’t be changed in place
- Provide security when you know data doesn’t need to change
- Sometimes you have to use them

Exercises

- What is the output for these operations?
- L = [1.0, 2, 'three', 0.8] T = ('A', 'b', 'C')
  - L[3]
  - T[1] = 'B'
- How could I…
  - Change the 'three' in L to 3?
  - Delete all but the first item in L?

Summary of built-in types



Operator precedence

- AKA order of operations

Operator	Description
x or y, lambda	Logical or, anonymous function
x and y	Logical and
not x	Logical negation
<, <=, >, >=, ==, is, is not, in, not in	Comparisons, identity tests, membership
x + y, x – y	Addition/concatenation, subtraction
x * y, x / y, x % y	Multiplication/repetition, division, modulo/format
-x	Unary negation
**	Exponentiation
x[i], x[i:j], x(…)	Indexing, slicing, function calls
(…), […], {…}, '…'	Tuple, list, dictionary, conversion to string

- Things lower in table are performed first
- Parenthesis always take precedence

Exercises

- How could we…
  - Calculate how many weeks are in 180 days?
  - Calculate the area of a circle given the diameter?
  - Given a list of [A, B, C], solve for the roots of $Ax^2 + Bx + C = 0$?
  - Calculate K_m and V_{max} given a pair of substrate concentration/enzyme activity data points?
    - Try with tuples: $([S]_1, v_1)$ and $([S]_2, v_2)$