BST281: Genomic Data Manipulation, Spring 2018

Wednesday 03: Python Loops, Conditionals, and Functions

Today’s lecture will focus on Python: completing our treatment of loops and conditionals and then proceeding to syntax for defining our own functions.

**Python: new material**

* The for loop revisited
  + Syntax for executing the same block of code multiple times.
  + In Python, the for loop performs a set of actions on each element of an “iterable” piece of data, often one of the built-in data collections (such as the list or dict).
  + range( ), enumerate( ), and sorted( ) are built-in helper functions for for loops.
  + .items( ) is a dictionary method to loop through (key, value) pairs.
* The if-else statement
  + Syntax for achieving “flow of control” in a program.
  + Do one thing if a given statement is True, do something else if is False.
  + Can sandwich one or more elif statements to build a “switch.”
* Logical statements
  + Use operators (>, <, <=, >=, ==, !=, in) to “compare” data.
  + **Note:** Know the difference between “=” and “==”.
  + in is a special Python operator: it tests for collection membership or substring relationship.
  + Can combine comparisons with logical operators and, or, not.
* The while loop
  + Syntax for performing an action (block of code) as long as a statement is True.
  + **Note:** If the statement is always True, a while loop will run forever.
* More ways to control flow
  + pass is Python’s way to say “do nothing.”
  + continue will immediately proceed to the next iteration of a loop.
  + break will exit a loop immediately.
* Functions
  + Defining functions provides a way to reuse pieces of code.
  + Functions act as a “pipe”: they take arguments (data), act on them, and return a result.
  + **Note:** Data are passed to functions “by reference.” Collection data (e.g. lists) passed to a function can be permanently changed by the actions of the function.
  + Keyword arguments provide defaults to a function that can be overridden.

# Suggested textbook reading

* Haddock & Dunn, Chapter 7
* Haddock & Dunn, Chapter 9
* **Note:** Haddock & Dunn introduce Python functions in Chapter 10 (p. 188-92), but their examples draw on material that we won’t see until later in this course.