Interpreter

• Use Python 3.x.x
  • Default python3 interpreter
  • Recommend ipython for interactive work

• White space sensitive
  • Indentation instead of braces and semicolons

• Most basic use: calculator
Variables

• Variables, identifiers
  • No declarations of type
  • Type deduced by interpreter during initialization
  • Since everything is an object, can do type introspection on variables
    • \texttt{dir(myvar)}
    • \texttt{type(myvar)}
    • \texttt{isinstance(myvar, sometype)}
Sequences

• Sequences
  • Linear data collections
    • Lists, tuples, and strings are most common
  • Lists are mutable, can alter in place
  • Strings and tuples are immutable, can’t change
Lists

• Mutable
• Create with [] or list()
• + overloaded for list concatenation
• Index from 0 (zero), can use negative numbers
• Slices give subsets of list, continuous or not
• Can mix types in same list
  • No separate containers for each type
Tuples

• Immutable
  • Otherwise very much like lists, but use parentheses instead of brackets.

• Handy for simulating multi-value returns
Strings

• Immutable
• Single or double quoted (triple quoted later)
  • There are no solitary char types
• Dozens of useful methods, see dir(‘a’)
Conditionals

• `if`, `else`, `elif`
• No parentheses around condition
• End with colon
• Indent condition body one level
Loops

• `while`, `for-in`
• No parentheses around condition
• End with colon
• Indent body one level
• While just as expected
• For more like Foreach in Java, Range-based for in C++11
  • Work directly with list elements, no loop counter `var`
Functions

• Declare with `def`
• No return type (deduced!)
  • Implies functions are objects
• No parameter types
  • If required, function must check
• Can return a tuple
  • Called ‘unpacking’