Python Quick-Reference Guide

Feature	Code	Comments
Basic math operators	`+ - * / **`	`**` raises to a power
Equals sign usage	`x = value` `x == value`	Sets `x` equal to `value` Checks whether `x` is equal to `value`
Basic print	`print("Text here", value)`	Prints text followed by the value of the given number or variable.
Formatted print	`print("The values are %3.2f and %d", %(x1, x2))`	Prints text with values embedded, formatted according to the given formatting characters.

Indexing & Slicing	`mylist[n]`
	
	`mylist[n:m:p]`
		Selects element with index n
		
		Selects list elements from $(n+1)^{th}$ to m^{th}, stepping by p

Ranges of values	`range(start, end, step)`
	
	`np.arange(start, end, step)`
	
	`np.linspace(start, end, number)`
		Create a *list* of values from `start` to `end - step`, separated by `step`.
		Create an *array* of values from `start` to `end - step`, separated by `step`.
		Create an *array* of values from `start` to `end`; `number` values in total.

"For" loop	`for i in range(start, end, step):` `    <code>`	

"If" statement	`if condition(s):`
	`    <code>`
	`elif condition(s):`
	`    <code>`
	`else:`
	`    <code>`
		Multiple conditions can be connected by "`or`" or "`and`." The `elif` and `else` are optional. Code blocks must be tab-indented.

"While" statement	`while condition(s):` `    <code>`	Multiple conditions can be connected by "`or`" or "`and`." Code blocks must be tab-indented.

Defining a function	`def myFunc(arguments):`
	`    <code>`
	`    return x, y`
		`arguments` are comma-separated. Use `return` to output values that can be used by other code.

Calling a function	`out1, out2 = myFunc(4, "Big", -0.36)`	The number of outputs must match the number of returned values

Importing functions or packages	`from numpy import sqrt`
	`from numpy import *`
	`import numpy as np`
		Import a single function
		Import an entire package
		Import a package with "nickname" (call a package function using `np.funcname`)

Numpy arrays	`np.array([1,2,3],'d')`
	`np.array([1,2,3],'i')`
	`np.array([[1,2],[3,4]],'d')`
	`np.zeros((n,m),'d')`
	`np.ones((n,m),'d')`
		Create 1 x n vector of floats
		Create 1 x n vector of integers
		Create 2 x 2 array of floats
		Create n x m array of `0.` Values
		Create n x m array of `1.` values

Altering lists	`myList.append(xyz)` `myList.remove(xyz)`	Adds “xyz” to end of list Removes element “xyz” from list
Plotting inside a notebook	`%matplotlib inline` `from matplotlib.pyplot import *`	