# Python Quick Reference

*OPERATOR PRECEDENCE IN EXPRESSIONS*

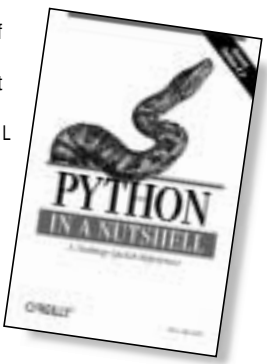
| Operator | Description | A |
|---|---|---|
| `expr,...` | String conversion | NA |
| {key:expr,...} | Dictionary creation | NA |
| [expr,...] | List creation | NA |
| (expr,...) | Tuple creation or simple parentheses | NA |
| f(expr,...) | Function call | L |
| x[index:index] | Slicing | L |
| x[index] | Indexing | L |
| x.attr | Attribute reference | L |
| x**y | Exponentiation (x to yth power) | R |
| ~x | Bitwise NOT | NA |
| +x, -x | Unary plus and minus | NA |
| x*y, x/y, x//y, x%y | Multiplication, division, remainder | L |
| x+y, x-y | Addition, subtraction | L |
| x<<y, x>>y | Left-shift, right-shift | L |
| x&y | Bitwise AND | L |
| x^y | Bitwise XOR | L |
| x\|y | Bitwise OR | L |
| x<y, x<=y, x>y, x>=y | Comparisons | C |
| x<>y, x!=y, x==y | Equality/inequality tests* | C |
| x is y, x is not y | Identity tests | C |
| x in y, x not in y | Membership tests | C |
| not x | Boolean NOT | NA |
| x and y | Boolean AND | L |
| x or y | Boolean OR | L |
| lambda arg,...: expr | Anonymous simple function | NA |

** x!=y and x<>y are the same inequality test (!= is the preferred form, <> obsolete)*

A – Associativity L – Left R – Right C – Chaining NA – Not associative

*LIST OBJECT METHODS*

| Operator | Description |
|---|---|
| L.count(x) | Returns the number of occurrences of x in L |
| L.index(x) | Returns the index of the first occurrence of x in L or raises an exception if L has no such item |
| L.append(x) | Appends x to the end of L |
| L.extend(l) | Appends all the items of list l to the end of L |
| L.insert(i,x) | Inserts x at index i in L |
| L.remove(x) | Removes the first occurrence of x from L |
| L.pop(i=-1) | Returns the value of the item at index i and removes it from L |
| L.reverse( ) | Reverses, in-place, the items of L |
| L.sort(f=cmp) | Sorts, in-place, the items of L, comparing items by f |




Excerpted from *Python in a Nutshell*

O'REILLY®

*www.oreilly.com*

# Python Quick Reference

*COMMON FILE OPERATIONS*

| Operation | Interpretation |
|---|---|
| output = open('/tmp/spam', 'w') | Create output file ('w' means write). |
| input = open('data', 'r') | Create input file ('r' means read). |
| S = input.read( ) | Read entire file into a single string. |
| S = input.read(N) | Read N bytes (1 or more). |
| S = input.readline( ) | Read next line (through end-line marker). |
| L = input.readlines( ) | Read entire file into list of line strings. |
| output.write(S) | Write string S into file. |
| output.writelines(L) | Write all line strings in list L into file. |
| output.close( ) | Manual close (done for you when file collected). |

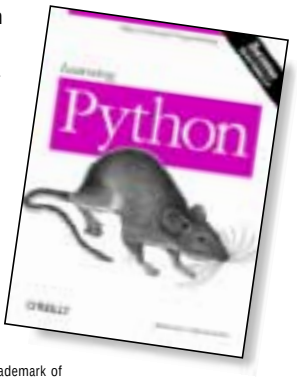
*COMMON DICTIONARY LITERALS AND OPERATIONS*

| Operation | Interpretation |
|---|---|
| D1 = { } | Empty dictionary |
| D2 = {'spam': 2, 'eggs': 3} | Two-item dictionary |
| D2['eggs'] | Indexing by key |
| D2.has_key('eggs'), 'eggs' in D2 | membership test |
| D2.keys( ), D2.values( ), D2.items( ) | lists of keys, values, items |
| D2.copy( ), D2.update(D1) | shallow copy, dict merging |
| D2.get(key, default=None) | "indexing" w/default value |
| len(D1) | Length (number stored entries) |
| D2[key] = 42 | Adding/changing, |
| del D2[key] | deleting |
| D4 = dict(zip(keyslist, valslist)) | Construction |

*COMMON TUPLE LITERALS AND OPERATIONS*

| Operation | Interpretation |
|---|---|
| ( ) | An empty tuple |
| T1 = (0,) | A one-item tuple (not an expression) |
| T2 = (0, 'Ni', 1.2, 3) | A four-item tuple |
| T2 = 0, 'Ni', 1.2, 3 | Another four-item tuple (same as prior line) |
| T1[i] | Indexing |
| T1[i:j] | slicing |
| len(t1) | length (number of items) |
| T1 + T2 | Concatenation |
| T2 * 3 | repetition |
| for x in T2 | Iteration |
| 3 in T2 | member-ship test |

Excerpted from *Learning Python,* 2nd Edition




O'REILLY®
*www.oreilly.com*