

Operators	File operations (file f) (cont)	Control flow (cont)
Assignment =	f.close() Close f	while (cond) : While loop <code>
Arithmetic +, -, *, /, %	When using: with open(path) as f: the file gets opened as f and closes after leaving the "with" statement	break Exit loop
Comparison >, >=, <, <=, ==, !=		continue Skip to next iteration
Logical not, and, or		
String operations (string s)	Base functions	Object-oriented
s.count(substring) Count occurrences)	int(), float(), str(), bool() ...	class Person: Class definition
s.find(substring) Index of first occurrence	len(data) Return length of data	x = Person(age, height) Object creation
s.join(sequence) Concatenate sequence	min(values), max(values)	x.age Filed access
s.split([deilimite r]) Split into list	pow(x, y, [z]) x to the power y [mod z]	x.birthday() Method access
List operations (list l, element e)	range(start, stop, [step]) Ordered list	List comprehensions
l.append(e) Add e	input(), print() Console input/ output	List comprehensions can be used to generate lists with the use of functions in just one line
l.remove Remove e	filter(function, iterable) Filter iterable	S = [x**2 for x in range(10)]
l.pop(e) Remove and return e	map(function, iterable) Map function onto iterable	M = [x for x in S if x % 2 == 0]
l.count(e) Count occurrences	id(object) Unique object ID	noprimes = [j for i in range(2, 8) for j in range(i*2, 50, i)]
l.reverse() Reverse l	round(n, [x]) Round n [x decimal places]	primes = [x for x in range(2, 50) if x not in noprimes]
l.sort() Sort l	create your own functions with: def functionname:	[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]
Dictionary operations (dict d, key k)	Control flow	Module Import
d.clear Clear d	if(cond): <code> else: <code> If-else statement	import module Imports a module
d.get(k) Return d[k]	if(cond): <code> elif(cond): <code> else: <code> If-elseif- else statement	import module as x Imports a module as x
d.keys() Return keys in d	for i in range([start], stop, [step]): <code> For loop over range	from module import submodule Imports specific submodule
d.values() Return values in d	for i in items: <code> For loop over iterable	
d.items() Return key-value pairs in d		
File operations (file f)		
f = open(path) Open file at path as f		
f.read() Read f		
f.readline() Read line from f		
f.readlines() Return list of lines in f		
f.write(s) Write s to f		

