

Operators

Assignment	=
Arithmetic	+, -, *, /, %
Comparison	>, >=, <, <=, ==, !=
Logical	not, and, or

String operations (string s)

s.count(s ubstring)	Count occurrences
s.find (sub string)	Index of first occurrence
s.join (sequence)	Concatenate sequence
s.split ([delimiter])	Split into list

List operations (list l, element e)

l.append(e)	Add e
l.remove	Remove e
l.pop(e)	Remove and return e
l.count(e)	Count occurrences
l.reverse()	Reverse l
l.sort()	Sort l

Dictionary operations (dict d, key k)

d.clear	Clear d
d.get(k)	Return d[k]
d.keys()	Return keys in d
d.values()	Return values in d
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
f.close()	Close f

When using:
with open(path) as f:
the file gets opened as f and closes after leaving the "with" statement

Base functions

int(), float(), str(), bool() ...

len(data)

min(values),
max(values)

pow(x, y, [z])

range(start, stop, [step])

input(), print()

filter(function, iterable)

map(function, iterable)

id(object)

round(n, [x])

create your own functions with:
def functionname:

Control flow

List comprehensions

List comprehensions can be used to generate lists

Return [x**2 for x in range(10)]

length of data
M = [x for x in S if x % 2 == 0]

primes = [j for i in range(2, 8)

primes = [x for x in range(2, 50)

Maximum 5, 7, 11, 13, 17, 19, 23, 2

Minimum

Maximum

Module Import

Import module

Ordered

list Import module as x

Console

input/

output module import submodule

Filter

iterable

Map

function

onto

iterable

Unique

object ID

Round n

[x

decimal

places]

<code>if(cond): <code> else: <code></code>	If-else statement
<code>if(cond): <code> elif(c ond): <code> else: <code></code>	If-elseif- else statement
<code>for i in range([st art], stop, [step]): <code></code>	For loop over range
<code>for i in items: <code></code>	For loop over iterable
<code>while(cond): <code></code>	While loop
<code>break</code>	Exit loop
<code>continue</code>	Skip to next iteration

Object-oriented

<code>class Person:</code>	Class definition
<code>x = Person (age, height)</code>	Object creation
<code>x.age</code>	Filed access
<code>x.birt hday()</code>	Method access



By **trashstuff**
cheatography.com/trashstuff/

Not published yet.
 Last updated 23rd February, 2017.
 Page 1 of 2.

Sponsored by **CrosswordCheats.com**
 Learn to solve cryptic crosswords!
<http://crosswordcheats.com>