Cheatography

Python3 Beginner Sheet Cheat Sheet

by Jespeer via cheatography.com/192168/cs/39953/

Fundamentals

```
my_int = 5
```

```
my_string = " Hello World"
```

```
my_oth er_ string = 'Hello ther
e'
```

Print to console: `print( " Hello Worl d! ")`

Collect input (string): `my_input = inpu t( " Write: ")`

Addition: `5 + 3`

Subtraction: `5 - 3`

Multiplication: `5 * 3`

Float division: `5 \ 3`

Integer division (floors): `5 \\ 3`

Modulo: `5 % 3`

Exponentiation: `5 ** 3`

True: `True`

False: `False`

Type conversion: `str(), int(), float( ), bool()`

Detect the type: `type()`

Equality of values: `a == b`

Equality of references: `a is b`

Conditional Logic

```
if
(boolean_condition_1):
        res ult_1
elif (boole an_ con -
dit ion_2):
        res ult_2
else:
        res ult _de -
fault
```

If one block has been executed, the others **cannot** be executed, even if their relative condition is `True`.

Looping

```
for i in range(5):
        pri nt(i) #
Prints 0 to 4
for i in ["Al pha ",
" Bet a"]:
        pri nt(i) #
Prints " Alp ha" and " -
Bet a"
j = 0
while (j < 3):
        pri nt(j) #
Prints 0 to 3
       j = j+1
```

You can loop through an iterable, which can be a range, a list, a tuple, a dictionary using `fo r` or through a condition, using `w hile`.

List Comprehension

```
# Squares of even
numbers less than 10
[i * i for i in
range(10) if i % 2 == 0]
# Multiple conditions
(odds become negative)
[i if i % 2 == 0 else -i
for i in range(10]
```

List Methods

Add an item to the end: `lst.ap - pen d(item)`

Add a list of items to the end: `lst .ex ten d(l ist _of _item s)`

Insert an item at an index: `lst.i n ser t(i ndex, item)`

Remove all items: `lst.cl ear( )`

Remove & return an item at an index: `lst.po p(i ndex)`

Remove the first item with given value: `lst.re mov e(val)`

List Methods (cont)

Removes an item at an index: `lst. de l(i ndex)`

Gives the index of a given value: `ls t.in dex (val, start, end)`

Gives the count of a given value: `l st.co unt (val)`

In-place reverses the list: `lst.re - verse()`

In-place sorts the list: `lst.sort()`

String aggregates: sep.join(lst)

Negative indices start from the end of the list, with `my_lis t[ -1]`. Using `list()` converts an iterable to a list.

Slicing

```
lst = [0, 5, 10, 15, 20]
# Generic
collec tio n[s tar t:e -
nd: step]
collec tion[3] # 15
collec tio n[1:] # [5,
10, 15, 20]
collec tio n[:2] # [0,
5]
collec tio n[::2] # [0,
10, 20]
collec tio n[::-1] #
[20, 15, 10, 5, 0]
```

Slicing can be applied to ordered collections. The start is **inclusive** and the end is **exclusive**.

Zip

Dictionary Methods