# Opensource.com: Python 3.7 Beginner's Cheat Sheet

BY NICHOLAS HUNT-WALKER

Python (v3.7.0) is a powerful programming language with a wide variety of uses. Even with Python's large community and diverse menu of extensions, plenty is built into the language itself that even a novice programmer can find useful. Here are a few of those built-in pieces:

Print a string, set of strings, or object representation(s) to the console.

```
>>> print('any string')
any string
>>> print('string one', 'string two')
string one string two
>>> print([1, 2.0, 'three', ['f','o','u','r']])
[1, 2.0, 'three', ['f', 'o', 'u', 'r']]
```

Return the number of items contained within a string or object.

```
>>> my_list = [41, 33, 48, 71, 60, 26]
>>> len(my_list)
6
```

View the documentation for any function or object that has it. To leave that view, hit the q key

```
>>> help(len)
Help on built-in function len in module
builtins:

len(obj, /)
    Return the number of items in a container.
(END)
```

Create an iterable of integers. You can either get sequential numbers starting at 0 up to (but not including) some higher number by providing one integer as an argument, set the bounds of your numbers by providing two integers, or set the bounds and the space between numbers with three integers.

```
>>> for number in range(4):
...     print(number)
0
1
2
3
>>> for number in range(3, 33, 10):
...     print(number)
3
13
23
```

You can't access these numbers by index like a list. You need to loop through the `range` to access them.

Access every item and that item's index within an iterable (e.g., a list) with `enumerate`.

```
>>> countries = ['Turks & Caicos', 'Grenada',
    'Vanuatu', 'Lebanon', 'Barbados']
>>> for idx, item in enumerate(countries):
... print(f'{idx} - {item}')
...
0 - Turks & Caicos
1 - Grenada
2 - Vanuatu
3 - Lebanon
4 - Barbados
```

Sort an iterable that contains objects that are all of the same type (as long as they can be compared to each other) without mutating the original object.

```
>>> sorted([64, 99, 69, 70, 53, 11, 42, 99, 5])
[5, 11, 42, 53, 64, 69, 70, 99, 99]
```

Open any file as text for reading or writing. By default the file is opened to read. If reading, the file must actually exist at the specified absolute or relative path.

```
>>> with open('some_text.txt') as f:
...     f.read()
...
'Whatever text was in the file'
>>> with open('writeable_file.csv', 'w') as f:
...     f.write('one,two,three')
...
```

Find out what type of object anything is with `type`.

```
>>> type([])
<class 'list'>
>>> type('random text')
<class 'str'>
```

Check whether an object is an instance of some built-in type or user-created `class` with `isinstance`.

```
>>> isinstance('potato', str)
True
>>> isinstance('potato', list)
False
```

 