

# Python Data Types

---

Wendley S. Silva

Agosto de 2020

Universidade Federal do Ceará



# Data types

- Numeric
- Boolean
- Sequence type
- Dictionary



# Numeric

Python identifies three types of numbers:

- **Integer:** Positive or negative whole numbers (without a fractional part)
- **Float:** Any real number with a floating point representation in which a fractional component is denoted by a decimal symbol or scientific notation
- **Complex number:** A number with a real and imaginary component represented as  $x+yj$ .  $x$  and  $y$  are floats and  $j$  is  $-1$  (square root of  $-1$  called an imaginary number)



# Boolean

Data with one of two built-in values **True** or **False**.

Notice that 'T' and 'F' are capital. true and false are not valid booleans and Python will throw an error for them.



# Sequence Type

A sequence is an ordered collection of similar or different data types. Python has the following built-in sequence data types:

- **String**: A string value is a collection of one or more characters put in single, double or triple quotes.
- **List**: A list object is an ordered collection of one or more data items, not necessarily of the same type, put in **square brackets**.
- **Tuple**: A Tuple object is an ordered collection of one or more data items, not necessarily of the same type, put in **parentheses**.



# Dictionary

A dictionary object is an unordered collection of data in a key:value pair form.

A collection of such pairs is enclosed in curly brackets.

For example: {1:"Steve", 2:"Bill", 3:"Ram", 4: "Farha"}



# type() function

```
>>> type(1234)
```

```
<class 'int'>
```

```
>>> type(55.50)
```

```
<class 'float'>
```

```
>>> type("hello")
```

```
<class 'str'>
```

```
>>> type([1,2,3,4])
```

```
<class 'list'>
```

```
>>> type((1,2,3,4))
```

```
<class 'tuple'>
```

```
>>> type({1:"one", 2:"two", 3:"three"})
```

```
<class 'dict'>
```



# Lists and tuples


---





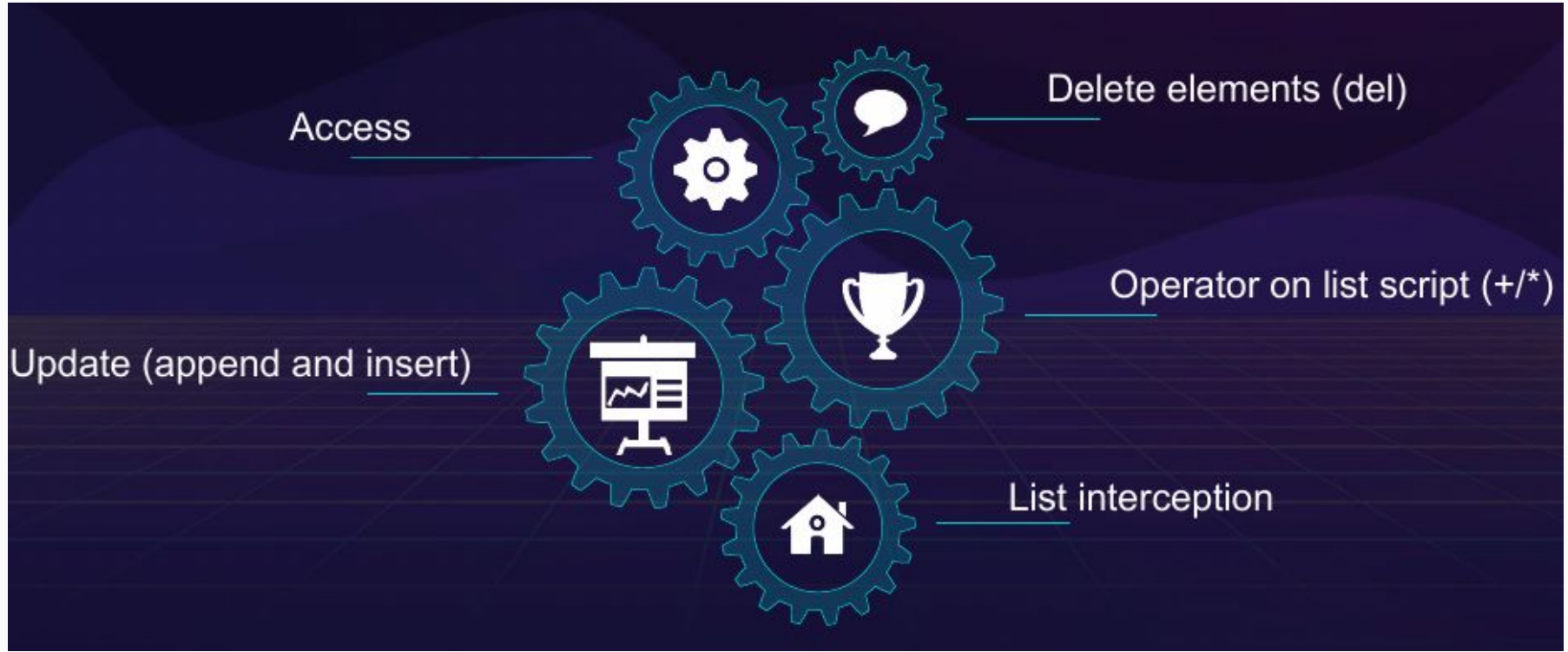
# List

- List is expressed using [ ].
- A list is an ordered set of elements that you can add and delete at any time.
- The element types in the list can be different. You can access each element in the list by index. The first digit of the index starts from 0, and the reverse index starts from -1.

	length = 5				
					
index	0	1	2	3	4
negative index	-5	-4	-3	-2	-1



# Common operations on Lists



# Functions of Python Lists



`cmp(list1,list2)`



`len(list)`



`max(list)`



`min(list)`



`list(seq)`

# Methods of Python Listing

1

`list.append(obj)`

2

`list.count(obj)`

3

`list.extend(seq)`

4

`list.index(obj)`

5

`list.insert(index, obj)`

6

`list.pop(obj=list[-1])`

7

`list.remove(obj)`

8

`list.sort([func])`

9

`list.reverse()`



# Tuples

- Tuple is expressed using ().
- A tuple is simple to create. You only need to add elements to parentheses and separate them with commas.
- Like a list, a tuple cannot be modified once initialized, and the elements need to be identified when a tuple is defined.
- A tuple does not have the append () or insert () method, nor can it be assigned to another element. It has the same fetching methods as a list.
- Because a tuple is unchangeable, the code is more secure. Therefore, if possible, use a tuple instead of a list.



# Common operations on Tuples

Access

Modify  
(tuple calculation)

Delete  
(del tuple)

Tuple operators  
(+,\*)

Tuple index and  
interception

No-close  
separator



# Embedded functions of Tuples



`cmp(tuple1, tuple2)`



`len(tuple)`



`max(tuple)`



`min(tuple)`



`tuple(seq)`

# Dictionaries

---



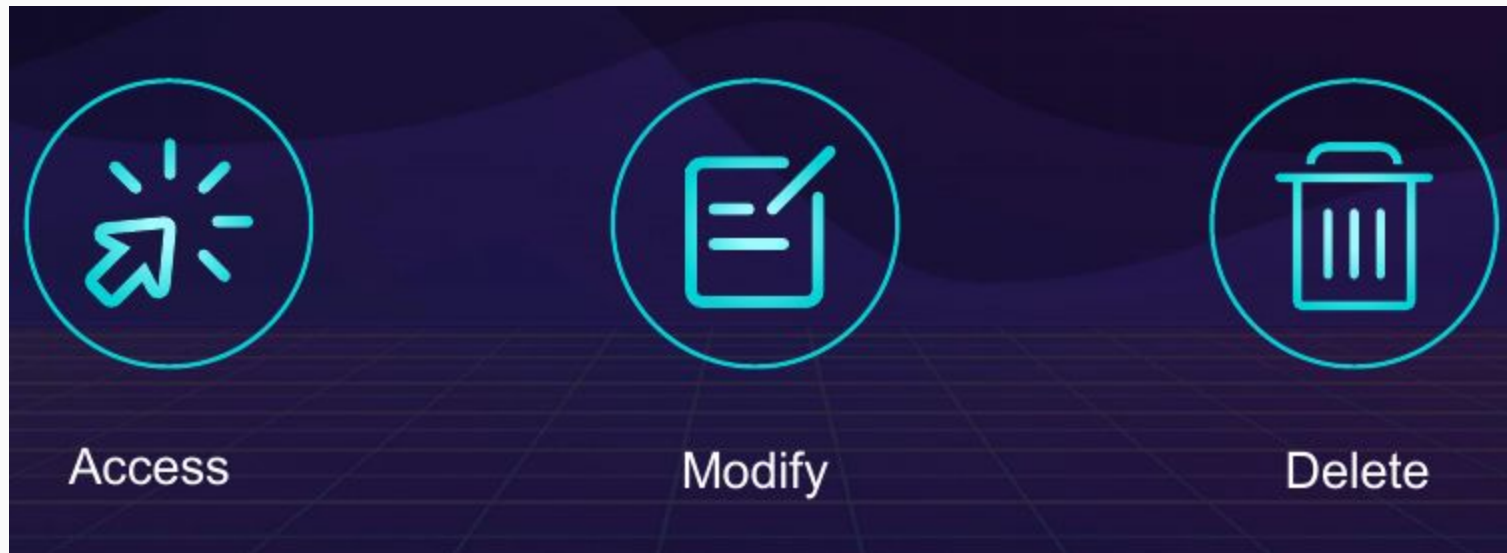


# Dictionaries

- A dictionary is another variable container model and can store any type of object.
- Each key value of the dictionary is separated with a colon ":" key value pairs are separated by a comma "," and the entire dictionary is included in the curly braces "{}".
- A dictionary has the following format
  - `d = {key1 : value1, key2 : value2 }`



# Python Dictionaries Operations



# Built-in functions of Dictionaries



`cmp(dict1, dict2)`



`len(dict)`



`str(dict)`



`type(variable)`

# Strings

---



# Definitions of a String

- In Python, a string is a sequence of 0 or more characters, and a string is one of several sequences built in Python.
- Python strings may be expressed using single quotes, double quotes and triple quotes, as well as escape characters, raw strings, and so on.

**name='JohnSmith'**

**name="Alice"**

**name=""""Bob"""**



# String formatting

- Python supports the output of formatted strings. Although a complex expression may be used, the most basic use is to insert a value into a string.
- String formatting in Python is accomplished by the string formatting operator (%), and its string conversion type table and its formatting operator auxiliary instructions are shown in the following tables.

Input: `print("My name is %s and age is %d !" %('AI', 63))`

Output: My name is AI and age is 63 !



# String formatting

Format	Description
%c	Character and its ASCII code
%s	String
%d	Signed integer (decimal)
%u	Unsigned integer (decimal)
%o	Unsigned integer (octal)
%x	Unsigned integer (hexadecimal)
%X	Unsigned integer (hexadecimal upper-case letters)
%e	Floating number (scientific counting method)
%E	Floating number (scientific counting method, E replacing e)
%f	Floating number (decimal point sign)
%g	Floating number (%e or %f, depending on a value)
%G	Floating number (similar to %g)
%p	Pointer (print memory address of a value using hexadecimal)



# String formatting

Symbol	Function
*	Defines width or precision of the decimal point.
-	Aligns to the left.
+	Displays + before a positive value.
<sp>	Displays space before a positive value.
#	Displays 0 before an octal number or 0x or 0X before a hexadecimal value (depending on whether x or X is used).
0	Adds 0 instead of default space before numbers.
%	%% outputs a single %.
(var)	Maps variables (dictionary arguments).
m.n	m means the minimum width and n means the number of digits after the decimal point.





# String operators

Python does not have dedicated Char type, a string is the sequence of a character and one character is a string with a length of 1. Python strings are not changeable and do not end with '\0'. and is stored in memory as follows:

	P	y	t	h	o	n
Superscript	0	1	2	3	4	5
Subscript	-6	-5	-4	-3	-2	-1



# String methods

```
>>> dir("")
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__doc__', '__eq__', '__format__',  
 '__ge__', '__getattribute__', '__getitem__', '__getnewargs__', '__getslice__', '__gt__',  
 '__hash__', '__init__', '__le__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__',  
 '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__rmod__', '__rmul__',  
 '__setattr__', '__sizeof__', '__str__', '__subclasshook__', '_formatter_field_name_split',  
 '_formatter_parser', 'capitalize', 'center', 'count', 'decode', 'encode', 'endswith',  
 'expandtabs', 'find', 'format', 'index', 'isalnum', 'isalpha', 'isdigit', 'islower', 'isspace',  
 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'partition', 'replace', 'rfind', 'rindex', 'rjust',  
 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title',  
 'translate', 'upper', 'zfill']
```



# String modules

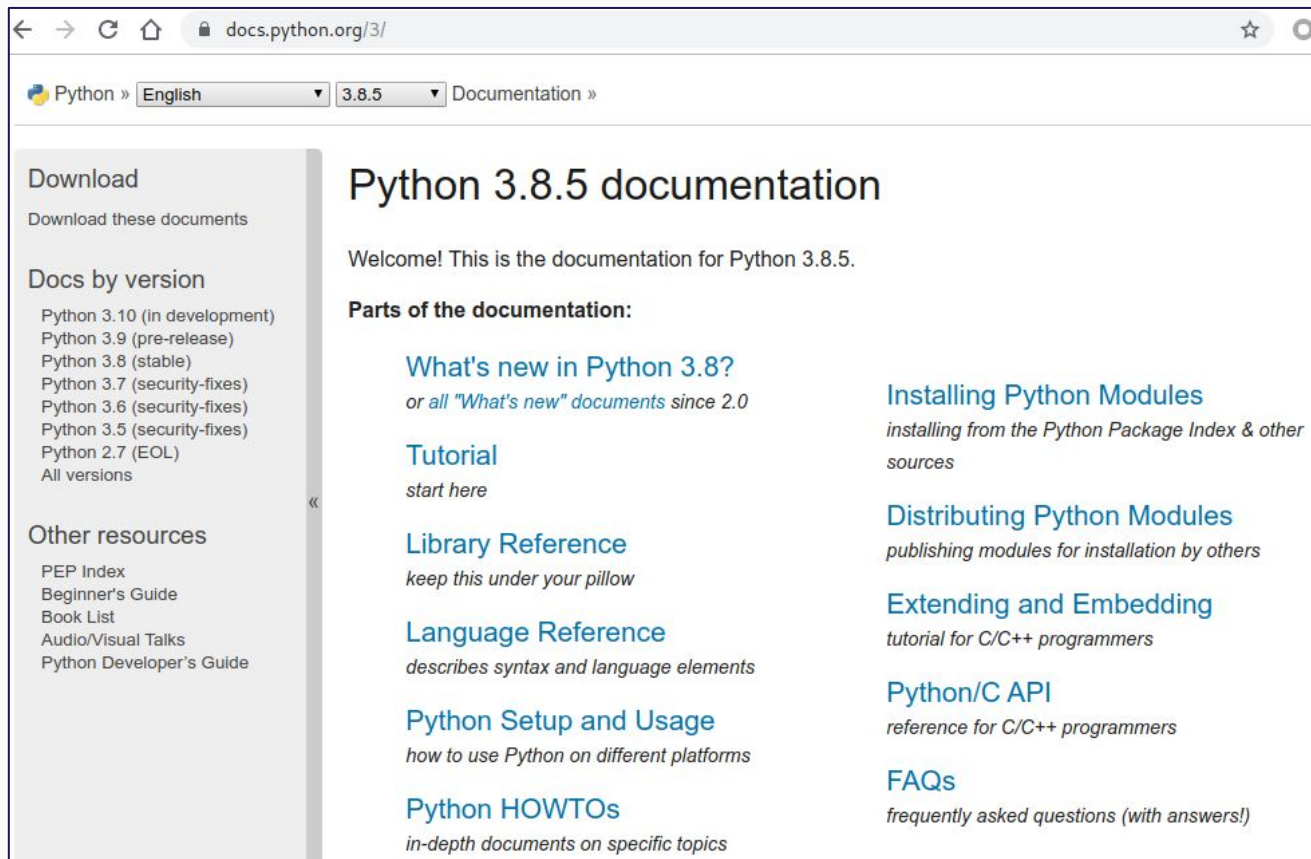
```
>>> import string
```

```
>>> dir(string)
```

```
['Formatter', 'Template', '_TemplateMetaclass', '__builtins__', '__doc__', '__file__',  
 '__name__', '__package__', '_float', '_idmap', '_idmapL', '_int', '_long', '_multimap',  
 '_re', 'ascii_letters', 'ascii_lowercase', 'ascii_uppercase', 'atof', 'atof_error', 'atoi',  
 'atoi_error', 'atol', 'atol_error', 'capitalize', 'capwords', 'center', 'count', 'digits',  
 'expandtabs', 'find', 'hexdigits', 'index', 'index_error', 'join', 'joinfields', 'letters', 'ljust',  
 'lower', 'lowercase', 'lstrip', 'maketrans', 'octdigits', 'printable', 'punctuation',  
 'replace', 'rfind', 'rindex', 'rjust', 'rsplit', 'rstrip', 'split', 'splitfields', 'strip', 'swapcase',  
 'translate', 'upper', 'uppercase', 'whitespace', 'zfill']
```



# Documentation



The screenshot shows the Python 3.8.5 documentation page. The browser address bar shows 'docs.python.org/3/'. The page has a sidebar on the left with navigation links and a main content area on the right. The sidebar includes 'Download', 'Docs by version', and 'Other resources'. The main content area has a large heading 'Python 3.8.5 documentation' and a list of links to various parts of the documentation, each with a brief description.

Python » English » 3.8.5 » Documentation »

## Download

Download these documents

## Docs by version

- Python 3.10 (in development)
- Python 3.9 (pre-release)
- Python 3.8 (stable)
- Python 3.7 (security-fixes)
- Python 3.6 (security-fixes)
- Python 3.5 (security-fixes)
- Python 2.7 (EOL)
- All versions

## Other resources

- PEP Index
- Beginner's Guide
- Book List
- Audio/Visual Talks
- Python Developer's Guide

## Python 3.8.5 documentation

Welcome! This is the documentation for Python 3.8.5.

### Parts of the documentation:

- [What's new in Python 3.8?](#)  
*or all "What's new" documents since 2.0*
- [Tutorial](#)  
*start here*
- [Library Reference](#)  
*keep this under your pillow*
- [Language Reference](#)  
*describes syntax and language elements*
- [Python Setup and Usage](#)  
*how to use Python on different platforms*
- [Python HOWTOs](#)  
*in-depth documents on specific topics*
- [Installing Python Modules](#)  
*installing from the Python Package Index & other sources*
- [Distributing Python Modules](#)  
*publishing modules for installation by others*
- [Extending and Embedding](#)  
*tutorial for C/C++ programmers*
- [Python/C API](#)  
*reference for C/C++ programmers*
- [FAQs](#)  
*frequently asked questions (with answers!)*



**Obrigado pela atenção**

