Python Data Structures Cheatsheet

Operators

Symbol	What it does	Example	
+	addition	5 + 5	
-	subtraction	5 - 5	
*	multiplication	5 * 5	
/	division	5 / 5	
**	exponent	5**5 (5 to the fifth power)	
%	modulus (remainder)	5 % 2	results in 3

==, is	Equals	`5 == 5`
		`9 is 9`
		`5 is 7`
			results in True
			results in True
			results in False

!=, is not	Not equals	`5 != 5`
		`5 is not 7`
			results in False
			results in True

>	Greater than	`5 > 6`
		`11 > 6.23`
			results in False
			results in True

>=	Greater than or equal to	`5 >= 4`	results in True
<	Less than	`4 < 5`	results in True
<=	Less than or equal to	`4 <= 5`	results in True

`in`	Test if an item is in a list, string, dictionary, or tuple	`5 in [3,4,5,6]`
		`7.77 in (2,3,9)`
			results in True
			results in False

Variable types

Variable Type	Description	Casting	Examples
integer	whole number	`int()`	• `5` • `-11` • `0`
float	decimal number	`float()`	• `9.57` • `-0.256` • `1.0`
string	ordered, immutable character container	`str()`	• `“this is a string”` • `“67”`
list	ordered, mutable container	`list()`	• `[1,2,3,4]` • `["hi", "bye", 9, -22.1]`
dictionary	unordered, mutable container (associative array)	`dict()`	• `{“key1”: “value1”, 8: 76, “key2”: 88}`
tuple	ordered, immutable container	`tuple()`	• `(4, 9)` • `("word1", "word2", "word3")`

Python Data Structures Cheatsheet

USEFUL STRING METHODS

Remember – because strings are immutable, calling these methods on them will NOT change the variable itself!

Method	Description	Example
`.upper()`	converts to upper case	`hi = "my string"` `hi.upper()` returns "MY STRING"

`.lower()`	converts to lower case	`hi = "My String"`
		`hi.lower()` `returns "my string"`

`.split()`	split a string on a value into a list	`hi = "comma,sep,vals"` `hi.split(",")` returns ['comma','sep','vals']
`.strip()`	removes leading/trailing whitespace/value	`hi = "my string"` `hi.strip()` returns "my string" (there was no leading/trailing whitespace!) `hi.strip("g")` returns "my strin"
`.count()`	count instances of a character in a string	`hi = "my letterful string"` `hi.count("t")` returns 3
`.replace()`	replace all/some instances of a character in a string	`hi = "silliness"` `hi.replace("s", "5")` returns "5illine55" `hi.replace("s", "5", 1)` returns "5illiness"

USEFUL LIST METHODS

Method	Description	Example
`.append()`	Add value to the end of a list	`my_list.append(5)`
`.insert()`	Add value to a specific index in a list	`my_list.insert(5, "index #5 will be this string")`
`.remove()`	Remove all occurrences of a particular value from a list	`my_list.remove(5)`
`.index()`	Determine the index of a particular list value	`my_list.index(5)`

USEFUL DICTIONARY METHODS

Method	Description	Example
`.keys()`	Return a list of all keys in a dictionary	`my_dict.keys()`
`.values()`	Return a list of all values in a dictionary	`my_dict.values()`
`.items()`	Return a list of (key,value) tuples from a dictionary	`my_dict.items()`

PYTHON INDEXING RULES

Syntax: `container[x:y:z]`

x = start index (inclusive); y = ending index (exclusive); z = step

Example: `mylist = [1,2,3,4,5,6,7,8,9]`

`mylist[3]` → `4` ; `mylist[3:6]` → `[4,5,6]` ; `mylist[3:8:2]` → `[4,6,8]`