

STRINGS

Strings	string = "words" or 'words'
connect	string1+string2
multiply	string*3
compare ('if' or 'if not')	string1 == string2 (or != or > or <)
check ('if' or 'if not')	'substring' in string
position	string[0] or string[-1] or string[x]
slicing	string[start:end:step]
	string[:3] or string[1:]
	string[::-1] inverted

Slicing strings

slicing:	string[start:end:step]
from start to end	string[2:5]
from start	string[:5]
to the end	string[2:]
jumping	string[2:5:2]
negative/inverted	string[::-1]

Modify strings

remove whitespaces before and after the string	string.strip()
or remove any 'characters'	string.strip('ch')
only before(left) the string	string.lstrip()
only after(right) the string	string.rstrip()
replace string	string.replace("old value", "new value", counter)

counter is how many occurrences from start, nothing is default and means every occurrence

use it to remove whitespaces inside the string	string.replace(" ", "")
--	-------------------------

split strings create list from strings	string.split(separator,maxsplit)
---	----------------------------------

separator is optional, whitespace is default	string.split(",")
--	-------------------

Modify strings (cont)

<i>maxsplit is optional, all occurrences is default(-1)</i>	string.split("-",1)
find string	string.find("value",start,end)
<i>find the first occurrence of a substring in a string</i>	string.find("substring")
<i>start and end are optional</i>	string.find("substring",3,10)
<i>returns the position of the substring</i>	
<i>index and find are the same, except that when false find returns -1 and index returns error</i>	string.index("substring")
lowercase	string.lower()
uppercase	string.upper()
capitalize first char upper	string.capitalize()
title first char of each word upper	string.title()
join values of a list/dict/tuple/ into string	' '.join(list)
count the occurrences of substring in a string	string.count('substring')

Format strings

str.format	'Hello, {} and {}'.format(string, string2)
f-string (3.6+)	f'Hello, {string} and {string2}'
<i>Integer numbers into strings*</i>	f'Hello, {string:}_'
'b' - binary	{string:b}
'c' - character (ASCII)	'Hello, {example:c}'.format(example = 'String', ...)
'd' - decimal	'Hello, {0:d} and {1}'.format(string,string2)
'o' - octal	{string:o}
'x' - hexadecimal, lowercase	'Hello, {0:x} and {0:X}'.format(string)



Format strings (cont)

'X' - hexadecimal, uppercase	{string:X}
'n' - number, decimal in local language OS	{string:n}
Floating-point	f'{integer:f}' (6 standard)
<i>round(a,2)</i>	f'{integer:.2f}' (2 decimal digits)
'e' ou 'E' - scientific notation (6 standard)	{:e}.format(999)
'E' - scientific notation (6 standard)	f'{999:.3E}'
'g' precisão >=1, round digits <i>p</i> to significant digit	{:g}.format(12.1231235843)
'n' - same as 'g', but local language OS	f'{12.1231235843:n}'
'%' - multiply the number *100 and '%' after	f'{0.05:%}'
<i>two digits before '.' and two after</i>	{:2.2%}.format(0.05)

Spacing

'<' left align / '^' center align / '>' right align	{0:<16}.format(string) / f.'{string:^16}' / {:>16}
number of digits	{:4}
<i>whitespaces if the string has less than 16..</i>	{string:16}
<i>choose char instead of whitespace</i>	{*:16}

COLLECTIONS

List	collection which is ordered and changeable. Allows duplicate members.
Tuple	collection which is ordered and unchangeable. Allows duplicate members.
Dictionary	collection which is ordered and changeable. No duplicate members.
Set	collection which is unordered, unchangeable, and unindexed. No duplicate members.

LISTS

Lists	list = ["string", "string2", integer, Bool]
<i>ordered, changeable, allow duplicates</i>	list = [item1, item2, item3]
Length	len(list)
Access <i>[start index included:end index not]</i>	list[1] / list[-1], list[2:5], list[:5]
Check if Item exists	if "item" in list:
Change items	list[3] = "new_value"
<i>range of items</i>	list[2:5] = "new_value1", "new_value2"
insert(index,value) <i>without replacing any item</i>	list.insert(2, "item")
append() <i>add item to the end</i>	list.append("item")
extend() <i>add items from other list</i>	list.extend(list2)
<i>works with tuples also</i>	list.extend(tuple)
remove() <i>remove first matching value</i>	list.remove("string")
pop() <i>remove specified index, and returns it</i>	list.pop(1)
del() <i>remove specified index</i>	list.del(3)
clear() <i>empties the list</i>	list.clear()
sort() <i>sort the list alphabetically</i>	list.sort()
<i>sort descending</i>	list.sort(reverse=True)
reverse() <i>order</i>	list.reverse()
copy() <i>make a copy</i>	list2 = list.copy()
<i>or use list()</i>	list3 = list(list2)
Concatenate Lists	list3 = list1 + list2
<i>or use extend()</i>	list.extend(list2)
count() <i>returns the number of items*</i>	list.count() / list.count('value')
index() <i>finds the item and return its index</i>	list.index('value')
min() / max() / sum()	list.min() list.max() list.sum()



LISTS (cont)

enumerate(index, value)	for i, v in enumerate(list): /n "{i} : {v}"
for loop	for x in list:
<i>through index</i>	for x in range(len(list)):
<i>while loop</i>	while x <= len(list): /n i+=1
<i>list comprehension</i>	[print[x] for x in list]

TUPLES

Tuples	tuple = ("value1", int, bool, "value1")
<i>are unordered, unchangeable, allow duplicates</i>	tuple = (item1,)
Lenght	len(tuple)
Access [start index included: end index not]	tuple[1] / tuple[-1], tuple[2:5], tuple[:5]
Check if Item exists	if "item" in tuple:
Change items <i>tuples are unchangeable</i>	list = list(tuple)
<i>convert tuple to list and back to tuple</i>	tuple = tuple(list)
Concatenate <i>add tuple to a tuple</i>	tuple1 + tuple2 / tuple1 += tuple2
Lists inside a tuple <i>are changeable</i>	tuple = (["value1", "value2"], item2, item3)
count() <i>returns the number of items</i>	tuple.count() / tuple.count('value')
index() <i>finds the item and return its index</i>	tuple.index('value')
min() / max() / sum()	tuple.min() tuple.max() tuple.sum()
enumerate(index, value)	for i, v in enumerate(tuple): /n "{i} : {v}"
for loop	for x in tuple:
<i>through index</i>	for x in range(len(tuple)):
<i>while loop</i>	while x <= len(tuple): /n i+=1
<i>list comprehension</i>	[print[x] for x in tuple]

DICTIONARIES

Dictionaries	dict = {"key": "value", "key2": "value2"}
<i>ordered, changeable, do not allow duplicates</i>	dict={"key1": bool, "key2": int, "key3": [list]}
<i>dict cannot have same keys</i>	
Lenght	len(dict)
Access <i>get the value of the "key"</i>	value1 = dict["key1"]
dict.get("key", "return if not found")	value4 = dict.get("key4", "Not found")
List of Keys	x = dict.keys()
Check if key exists	if "keys" in dict:
<i>if values exists</i>	if "value1" in dict.values()
Change values of a key	dict["key"] = value
<i>using update()</i>	dict.update({'key': 'value'})
Add	dict["key"] = value
<i>also can use update()</i>	dict.update({'key': 'value'})
Remove <i>.pop or popitem(removes the last key inserted)</i>	dict.pop("key") / dict.popitem()
<i>using del</i>	del dict["key"]
<i>del can delete the dictionary completely</i>	del dict
clear <i>empties the dictionary</i>	dict.clear() or dict = {}
Copy	dict = dict2 / dict2 = dict.copy() / dict2 = dict(dict)
items()	dict.items()
keys()	dict.keys()
values()	dict.values()
for loop	for keys in dict:
<i>keys</i>	for keys in dict.keys():
<i>values</i>	for values in dict.values():
<i>keys and values</i>	for keys, values in dict.items()

