



# L#9. Strings

Ago 2017

<https://docs.python.org/2.5/lib/typesseq.html>





# Sequences for this course

```
import numpy as np
# Strings:
S='LEONARDO DA VINCI'
print(S)
# Lists:
L=[3,4.5,7+3j,'Leonardo',True,[1,2,3]]; LL=[1,2,3,4,5]
print(L);
# Tuples—same as Lists but immutable
T=(3,4.5,7+3j,(1,2,3),'Leonardo',True,[1,2,3])
print(T)
# Arrays
A=np.array([1,2,3,4,5])
print(A)
```



```
LEONARDO DA VINCI
[3, 4.5, (7+3j), 'Leonardo', True, [1, 2, 3]]
(3, 4.5, (7+3j), (1, 2, 3), 'Leonardo', True, [1, 2, 3])
[1 2 3 4 5]
```



# Strings

String literals are written in single or double quotes, e.g.: 'Hello World', "Hello World". The string objects are immutable (see optional slides at the end).

```
s='HELLO WORLD'      # s=input("Enter Greeting"), user: HELLO WORLD
print(type(s))      # str
print(len(s))        # 11, len is short for length (longitud de la cadena de caracteres)
print(s)              # HELLO WORLD
```

```
print(s[0])          # H
print(s[6])           # W
print(s[-1])          # D
```

index	0	1	2	3	4	5	6	7	8	9	10
character	'H'	'E'	'L'	'L'	'O'	' '	'W'	'O'	'R'	'L'	'D'
index	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1



# Slicing



index	0	1	2	3	4	5	6	7	8	9	10
character	'H'	'E'	'L'	'L'	'O'	' '	'W'	'O'	'R'	'L'	'D'
index	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

- Extract part of a string using the syntax [start:stop:step] which target characters from index **start** until index **stop** (exclusive) in increments of **steps**. The character at index **stop** is not included.
- Omitted index: if we omit **start**, it defaults to (first) index (i.e., **zero**). If we omit **stop**, it defaults to (**last**) index in the sequence. If we omit **step**, it defaults to (**+1**).

```
s="HELLO WORLD"
```

```
(1) print(s[0:5:1])
```

HELLO

```
(2) print(s[0:5])
```

HELLO # step defaults to 1, indexes: 0,1,2,3,4

```
(3) print(s[::2])
```

HLOWRD # start=0, stop=11, step=2

```
(4) print(s[6:])
```

WORLD # start=6, stop=11, step=1



Examples and quizzes in file: string04.py



## Slicing Step

index	0	1	2	3	4	5	6	7	8	9	10
character	'H'	'E'	'L'	'L'	'O'	' '	'W'	'O'	'R'	'L'	'D'
index	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

- One last word:
  - If step is positive, you go this direction: ➡
  - If step is negative, you go this direction: ⬅

```
s="HELLO WORLD"
```

```
print(s[0:5:1])           # HELLO
```

# While slicing, we want to reverse the order as: OLLEH. [Three solutions:](#)

```
print(s[4:-12:-1]; print(s[-7:-12:-1], print(s[4::-1]) # OLLEH
```



# Slicing

index	0	1	2	3	4	5	6	7	8	9	10
character	'H'	'E'	'L'	'L'	'O'	' '	'W'	'O'	'R'	'L'	'D'
index	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1



A bit more slicing:

```
s="HELLO WORLD"
```

```
print(s[:5:1])
```

HELLO

```
print(s[:5])
```

HELLO

```
print(s[:])
```

HELLO WORLD

# which is the same as print(s)

```
print(s[::])
```

HELLO WORLD

# which is the same as print(s)





# Strings: Basic Operations

code [File: string03.py]

output

```
# String literals can span multiple lines: triple quotes
print(""" Cuando hay una tormenta
los pajaritos se esconde,
pero las aguilas vuelan mas alto. """)
```

```
Cuando hay una tormenta
los pajaritos se esconde,
pero las aguilas vuelan mas alto.
# Mahatma Gandhi
```

```
# A plus (+) concatenates strings
print("Sa" + "cu" + "dio")
```

```
Sacudio
```

```
# Iterate
for x in "Hola Mundo": print(x, end="")
```

```
Hola Mundo
```

```
# Multiply
print('HOLA'*5)
```

```
HOLAHOLAHOLAHOLAHOLA
```

```
# Replace a substring in a string with something else
s="cocodrile"; s2=s.replace("drile","nut")
print(s2)
```

```
coconut
```



Ref: [https://www.w3schools.com/python/python\\_ref\\_string.asp](https://www.w3schools.com/python/python_ref_string.asp)



# Strings: Basic Operations

Operation	Result	Notes
<code>x in "Mexico"</code>	True	a member
<code>x not in "San Juan"</code>	True	not a member
<code>min("abcdefgh")</code>	a	smallest item of "abcdefgh"
<code>max("abcdefgh")</code>	h	largest item of "abcdefgh"
<code>print("PONCE"[0])</code>	P	first element
<code>print("MEXICO"[-1])</code>	O	last element
<code>print("PONCE"[0:3])</code>	PON	slice of first 3 elements
<code>s='MEXICO'</code> <code>print(s.replace('X',''))</code> <code>print(s)</code>	MEICO MEXICO	removes X







# Strings objects are immutable

File: strings02.py.

```
a = "co"           # a now points to "co"  
b = a              # b points to the same "co" that a points to  
a = a + a          # a points to the new string "coco", but b still points to the old "co"
```

```
print(a)           coco  
print(b)           co
```

The string objects themselves are immutable.

The variable, a, which points to the string, is mutable.

```
c = a + b + "drilo"  
print(c)           cococodrilo  
c[0]="C"           Error, bla, bla, bla
```



# THE END