4 Cadenas.

Objetivo. Explicar el concepto de variable, etiqueta, objetos y como se usan mediante algunos ejemplos.

Funciones de Python: - `print()`, `type()`, `id()`, `chr()`, `ord()`, `del()`

4.1 Definición de cadenas

Para definir una cadena se utilizan comillas simples `'`, comillas dobles `"` o comillas triples `"""` o `'''`.

```
        simples = 'este es un ejemplo usando \' \' '
        print(simples)

        dobles = "este es un ejemplo usando \" \" "
        print(dobles)

        triples1 = '''este es un ejemplo usando \''' \''' '''
        print(triples1)

        triples2 = """este es un ejemplo usando \""" \""" """
        print(triples2)
```

```
este es un ejemplo usando ' '
este es un ejemplo usando " "
este es un ejemplo usando ''' '''
este es un ejemplo usando """ """
```

Observa que para poder imprimir `'` dentro de una cadena definida con `' '` es necesario usar el caracter `\` antes de `'` para que se imprima correctamente. Lo mismo sucede en los otros ejemplos.

Es posible imprimir `'` sin usar el caracter `\` si la cadena se define con `"` y viceversa, veamos unos ejemplos:

```
        # La cadena puede tener ' dentro de " ... "
        poema = "Enjoy the moments now, because they don't last forever"
        print(poema)
```

```
Enjoy the moments now, because they don't last forever
```

```
        # La cadena puede tener " dentro de ' ... '
        titulo = 'Python "pythonico" '
        print(titulo)
```

```
Python "pythonico"
```

```
# La cadena puede tener " y ' dentro de ''' ... '''
queja = """
Desde muy niño
tuve que "interrumpir" 'mi' educación
para ir a la escuela
"""
print(queja)
```

```
Desde muy niño
tuve que "interrumpir" 'mi' educación
para ir a la escuela
```

```
# La cadena puede tener " y ' dentro de """ ... """
queja = """
Desde muy niño
tuve que "interrumpir" 'mi' educación
para ir a la escuela
"""
print(queja)
```

```
Desde muy niño
tuve que "interrumpir" 'mi' educación
para ir a la escuela
```

4.2 Indexación de las cadenas.

La indexación de las cadenas permite acceder a diferentes elementos, o rangos de elementos, de una cadena.

- Todos los elementos de una cadena se numeran empezando en `0` y terminando en `N`, el cual representa el último elemento de la cadena.
- También se pueden usar índices negativos donde `-1` representa el último elemento y `-(N+1)` el primer elemento.

Veamos la siguiente tabla:

cadena :	M	u	r	c	i	é	l	a	g	o
índice +:	0	1	2	3	4	5	6	7	8	9
índice -:	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
ejemplo = 'Murciélago'
```

```
ejemplo[0]
```

```
'M'
```

```
ejemplo[5]
```

```
'é'
```

```
ejemplo[9]
```

```
'o'
```

```
len(ejemplo) # Longitud total de la cadena
```

```
10
```

```
ejemplo[-1]
```

```
'o'
```

```
ejemplo[-5]
```

```
'é'
```

```
ejemplo[-10]
```

```
'M'
```

4.3 Inmutabilidad de las cadenas

Los elementos de las cadenas no se pueden modificar:

```
ejemplo[5] = "e"
```

```
TypeError: 'str' object does not support item assignment
```

```
cadena='''
esta es una
oración
larga
'''
```

```
print(type(cadena))
```

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

```
len(cadena)
```

```
27
```

```
cadena[0]
```

```
'\n'
```

```
cadena[-1]
```

```
'\n'
```

```
cadena[5] = 'h'
```

```
TypeError: 'str' object does not support item assignment
```

4.4 Acceso a porciones de las cadenas (*slicing*)

Se puede obtener una subcadena a partir de la cadena original. La sintaxis es la siguiente:

```
cadena[Start:End:Stride]
```

Start :Índice del primer caracter para formar la subcadena.

End : Índice (menos uno) que indica el caracter final de la subcadena.

Stride: Salto entre elementos.

```
ejemplo[:] # Cadena completa
```

```
'Murciélago'
```

```
ejemplo[0:5] # Elementos del 0 al 4
```

```
'Murci'
```

```
ejemplo[::2] # Todos los elementos, con saltos de 2
```

```
'Mrilg'
```

```
ejemplo[1:8:2] # Los elementos de 1 a 7, con saltos de 2
```

```
'ucéa'
```

```
ejemplo[::-1] # La cadena en reversa
```

```
'ogaléicruM'
```

4.5 Operaciones básicas con cadenas

Los operadores: `+` y `*` están definidos para las cadenas.

```
'Luis' + ' ' + 'Miguel' # Concatenación
```

```
'Luis Miguel'
```

```
'ABC' * 3 # Repetición
```

```
'ABCABCABC'
```

4.6 Funciones aplicables sobre las cadenas

Existen métodos definidos que se pueden aplicar a las cadenas. Véase Common string operations para más información.

```
ejemplo = 'murcielago'
```

```
ejemplo.capitalize()
```

```
'Murcielago'
```

```
print(ejemplo)
print(ejemplo.center(20,'-'))
print(ejemplo.upper())
print(ejemplo.find('e'))
print(ejemplo.count('g'))
print(ejemplo.isprintable())
```

```
murcielago
-----murcielago-----
MURCIELAGO
5
1
True
```

4.7 Construcción de cadenas con variables

```
edad = 15
nombre = 'Pedro'
apellido = 'Páramo'
peso = 70.5
```

Concatenación y casting.

```
datos = nombre + apellido + 'tiene' + str(15) + 'años y pesa ' + str(70.5)
datos
```

```
'PedroPáramotiene15años y pesa 70.5'
```

Método `format()`

```
datos = '{} {} tiene {} años y pesa {}'.format(nombre, apellido, edad, peso)
datos
```

```
'Pedro Páramo tiene 15 años y pesa 70.5'
```

Cadenas formateadas (*f-string*, *formatted string literals*)

```
datos = f'{nombre} {apellido} tiene {edad} años y pesa {peso}'
datos
```

```
'Pedro Páramo tiene 15 años y pesa 70.5'
```