PROGRAMACIÓN DE INTELIGENCIA ARTIFICIAL

PYTHON

```
>>> cadena = ("Sr\\Sra\nHola")
>>> print(cadena)
Sr\Sra
Hola
>>> len(cadena)
11
```

```
>>> cad = ''
>>> len(cad)
0
```

```
>>> cadena = '''1234
...     5678
... 90'''
>>> len(cadena)
12
```

```
>>> cadena = 'perico'
>>> 'b' in cadena
True
>>> 'b' not in cadena
False
```

```
str1 = 'a'
str2 = 'b'

print(str1 + str2) # Operador + -> Concatena
print(str2 + str1)
print(5 * 'a')     # Operador * -> Replica
print('b' * 4)
```

‘ab’
‘ba’
‘aaaaa’
‘bbbb’

```
# función ord()

char_1 = 'a'
char_2 = ' '  # space

print(ord(char_1))
print(ord(char_2))
```

97
32

chr(ord(x)) == x
ord(chr(x)) == x

```
# función chr()

print(chr(97))
print(chr(945))
```

a
α

```
# Indexando cadenas

string = 'Ahora'

for x in range(len(string)):
    print(string[x], end='-')
```

A-h-o-r-a-

```
# Iterando en una cadena

string = 'Ahora'

for char in string:
    print(char, end=' ')
```

```
# Porciones - Rebanadas
# Slices

alpha = "abdefg"

print(alpha[1:3])
print(alpha[3:])
print(alpha[:3])
print(alpha[3:-2])
print(alpha[-3:4])
print(alpha[::2])
print(alpha[1::2])
```

‘bd’
‘efg’
‘abd’
‘e’
‘e’
‘adf’
‘beg’

PROGRAMACIÓN DE INTELIGENCIA ARTIFICIAL

PYTHON

```
# función min()
# devuelve el caracter de menor ordinal en la cadena
# mayúsculas < minúsculas

print(min('''PalomitA'''))
print('[', min("Palo mitA"), ']')
```

A

[ ]

```
# función max()
# devuelve el caracter de mayor ordinal en la cadena

print(max('''PalomitA'''))
print('[', max("Palo mitA"), ']')
```

t

[t]

```
# método count()
# devuelve el número de apariciones
# del parámetro en la cadena

print("abracadabra".count("b"))
print('abracadabra'.count("da"))
```

2

1

Se comparan carácter a carácter

```
# método index()
# devuelve el índice de la primera ocurrencia
# del parámetro en la cadena


print("CcAaBbEeZzAa".index("b"))
print("CcAaBbEeZzAa".index("Z"))
print("CcAaBbEeZzAa".index("A"))
```

5

8

2

```
# función list()
# crea una nueva lista con los caracteres de la cadena

print(list("abcabc"))
```

['a','b','c','a','b','c']

Las cadenas son inmutables

No está permitido

- *del cadena[0]*
- *cadena.append('z')*
- *cadena.insert(1,'a')*

PROGRAMACIÓN DE INTELIGENCIA ARTIFICIAL

PYTHON

Formateo de cadenas

Los strings permiten técnicas avanzadas de formato mediante la función `.format()`

{} reemplaza el campo por el parámetro en .format() secuencialmente

```
>>> print("Mi nombre es {}, tengo {} años y vivo en {}".format ("Ana", 20, "París"))
Mi nombre es Ana, tengo 20 años y vivo en París
```

{n} reemplaza el campo por el enésimo parámetro en `.format()`.

```
>>> print("Mi nombre es {2}, tengo {1} años y vivo en {0}".format ("París", 20, "Eva"))
Mi nombre es Eva, tengo 20 años y vivo en París
```

En los marcadores de posición se puede añadir un tipo de formato, utilizando los «:»

```
>>> print("Su precio es {:.2f} Euros".format(3.1))
Su precio es 3.10 Euros
```

formatea el parámetro como un valor flotante de dos posiciones decimales

```
>>> print("La {1} cuesta {0:06.2f} euros".format(5.111111, "camisa"))
La camisa cuesta 005.11 euros
```

formatea el parámetro número 0
como un valor flotante de dos posiciones decimales
utilizando al menos 6 cifras (contando el punto decimal)

PROGRAMACIÓN DE INTELIGENCIA ARTIFICIAL

PYTHON

Formateo de cadenas

Python 3 permite las **`f-strings.`** Para especificar una cadena como ***f-string***, se coloca una f delante de la cadena literal y se agregan variables u otras expresiones escribiéndolas entre llaves {} .
Permite formatear valores de la misma forma que el método format()

```
>>> nombre="Ana"
>>> edad=20
>>> txt=f"Me llamo {nombre} y tengo {edad} años"
>>> print(txt)
Me llamo Ana y tengo 20 años
```

```
>>> nombre="Eva"
>>> altura=1.8
>>> texto=f"Me llamo {nombre} y mido {altura:.2f} metros"
>>> print(texto)
Me llamo Eva y mido 1.80 metros
```

```
def funcion():
    return 29

print(f"Su valor es {funcion()}")
Su valor es 29
```

PROGRAMACIÓN DE INTELIGENCIA ARTIFICIAL

PYTHON

Lista de los métodos disponibles de la clase string

- ⇨ capitalize(): cambia el primer carácter de la cadena a mayúsculas y el resto de letras a minúsculas.
- ⇨ center(): centra la cadena dentro de una longitud conocida.
- ⇨ count(): cuenta las ocurrencias de un carácter dado.
- ⇨ join(): une todos los elementos de un iterable en una cadena.
- ⇨ lower(): convierte todas las letras de la cadena en minúsculas.
- ⇨ lstrip(): elimina los caracteres en blanco al principio de la cadena.
- ⇨ replace(): reemplaza una subcadena dada con otra.
- ⇨ rfind(): encuentra la última aparición de una subcadena
- ⇨ rstrip(): elimina los caracteres en blanco al final de la cadena.
- ⇨ split(): divide la cadena en una lista de subcadenas usando un delimitador dado.
- ⇨ strip(): elimina los espacios en blanco iniciales y finales.
- ⇨ swapcase(): intercambia las mayúsculas y minúsculas de las letras.
- ⇨ title(): hace que la primera letra de cada palabra sea mayúscula.
- ⇨ upper(): convierte todas las letras de la cadena a mayúsculas.

- ⇨ endswith(): ¿termina con una subcadena determinada?
- ⇨ isalnum(): ¿consta solo de letras y dígitos?
- ⇨ isalpha(): ¿solo de letras?
- ⇨ islower(): ¿consta solo de letras minúsculas?
- ⇨ isspace(): ¿consta solo de espacios en blanco?
- ⇨ isupper(): ¿consta solo de letras mayúsculas?
- ⇨ startswith(): ¿empieza por una subcadena determinada?