

Programación con python

For Loop

For Loop

El bucle for en Python.

Sirve para repetir instrucciones un número definido de veces.

Recorre elementos de una secuencia (listas, tuplas, strings, range, etc.).

Sintaxis:

```
for variable in secuencia:  
    # bloque de código
```

For Loop

```
for letra in "Hola":  
    print(letra)
```

Salida:

H

o

l

a

For Loop - range

`range(inicio, fin, paso)` genera una secuencia de números.

Muy útil cuando queremos repetir algo varias veces.

```
for i in range(10):  
    print(i)
```

For Loop - range

```
for i in range(1, 11, 2):  
    print(i)
```

Salida:

```
# 1  
# 3  
# 5  
# 7  
# 9
```

inicio: primer número (incluido).

fin: hasta dónde llega (excluido).

paso: de cuánto en cuánto aumenta.

For Loop - iteraciones sobre colecciones

```
lst = [1, 2, 3, 4, 5, True, False]

for i in range(7):
    print(lst[i])
```

For Loop - iteraciones sobre colecciones

```
lst = [1, 2, 3, 4, 5, True]

for i in range(len(lst)):
    print(lst[i])
```

For Loop - iteraciones sobre colecciones

```
for element in lst:  
    print(element)
```


For Loop - iteraciones sobre colecciones

```
for i, element in enumerate(lst):  
    print(i, element)
```

For Loop - iteraciones sobre colecciones

```
tup = (2, 3, 4, "hello", "tim", True)
```

```
for i in range(len(tup)):
    element = tup[i]
    print(element)
```

```
for element in tup:
    print(element)
```

```
for i, element in enumerate(tup):
    print(i, element)
```

For Loop - iteraciones sobre colecciones

```
s = "my string"

for i in range(len(s)):
    print(s[i])
```

For Loop - Break

La instrucción `break` se usa dentro de un bucle (`for` o `while`) para detenerlo inmediatamente, sin importar si todavía quedan más elementos por recorrer o más iteraciones por hacer.

`break` corta el ciclo en el momento en que se cumple la condición indicada.

For Loop - Break

```
lst = [1, 2, 3, 3, 4, 4, 2, 1, 2]

for num in lst:
    if num == 4:
        break

    print(num)

print("Done")
```

For Loop - continue

La instrucción continue se usa dentro de un bucle (for o while) para saltar la iteración actual y continuar directamente con la siguiente.

continue no detiene el ciclo completo, solo evita ejecutar el código que queda en esa vuelta y pasa a la siguiente iteración.

For Loop - continue

```
lst = [1, 2, 3, 3, 4, 4, 2, 1, 2]

for num in lst:
    if num == 4:
        continue

    print(num)

print("Done")
```

Actividad

1. Suma de números hasta cierto límite

Recorre la lista:

`lst = [2, 4, 6, 8, 10, 12, 14]`

Suma los números hasta que aparezca un 10. Usa `break` y muestra la suma.

Salida esperada: 20 (2 + 4 + 6 + 8).

Actividad

2. Saltando múltiplos de 3

Recorre la lista:

`lst = [1, 2, 3, 4, 5, 6, 7, 8, 9]`

Imprime todos los números excepto los múltiplos de 3

Actividad

Salida esperada:

1
2
4
5
7
8

Actividad

3. Contar vocales en una cadena

Pide al usuario una palabra y cuenta cuántas vocales (a, e, i, o, u) tiene usando un for.

Si el usuario escribe "python", la salida debe ser:

Número de vocales: 1

Actividad

4. Tabla de multiplicar

Usa un bucle for con range para mostrar la tabla de multiplicar del 5, del 1 al 10.

$$5 \times 1 = 5$$

$$5 \times 2 = 10$$

...

$$5 \times 10 = 50$$

Actividad

5. Encontrar el mayor número

Dada la lista:

`lst = [15, 22, 8, 35, 19, 50, 41]`

Usa un bucle for para recorrer los elementos y encontrar el número más grande (sin usar `max()`).

Salida esperada:

El mayor número es: 50