Módulo 4.2: Iteradores FOR

Prof. Carlos Cedeño

¿Qué es un Bucle `for`?

Imagina que tienes que saludar a cinco personas. Podrías escribir `print("Hola")` cinco veces, pero ¿qué pasaría si fueran cien o mil personas? Sería muy ineficiente.

El bucle `for` automatiza esta tarea. Su función es **iterar** sobre una secuencia de elementos. "Iterar" es simplemente una forma elegante de decir "recorrer uno por uno".

La sintaxis básica es la siguiente:

```
for variable_temporal in secuencia:
    # Bloque de código que se repetirá
    # Este código se ejecuta para cada elemento en la secuencia
    print(variable_temporal)
```

- `for` : Es la palabra clave que inicia el bucle.
- `variable_temporal` : Es una variable que tomará el valor de cada elemento de la secuencia en cada repetición (o "iteración"). Puedes nombrarla como quieras (`x` , `elemento` , `nombre` , `numero` , etc.).
- `in` : Es la palabra clave que conecta la variable temporal con la secuencia.
- `secuencia` : Es el conjunto de elementos que queremos recorrer. Puede ser una lista, un rango de números, un texto y mucho más.
- `:` : Los dos puntos son cruciales. Indican el inicio del bloque de código que se va a repetir.
- **Bloque de código indentado**: Todo el código que esté con sangría (generalmente 4 espacios) debajo del `for` es lo que se ejecutará en cada iteración.

Formas de Usar el Bucle `for`

1. Recorriendo un Rango de Números con `range()`

La forma más común de empezar a usar `for` es con la función `range()`. Esta función genera una secuencia de números que el bucle `for` puede recorrer.

a) `range(stop)`: Desde 0 hasta un número antes del final.

Si le das un solo número (por ejemplo, 5), `range()` generará números desde 0 hasta el número *anterior* al que indicaste.

Ejemplo: Imprimir los números del 0 al 4.

```
print("Contando del 0 al 4:")
for numero in range(5):
    print(numero)
```

Salida:

```
Contando del 0 al 4:
0
1
2
3
4
```

b) `range(start, stop)`: Definiendo un inicio y un fin.

Puedes especificar desde qué número empezar y hasta cuál llegar (sin incluirlo).

Ejemplo: Imprimir los números del 2 al 6.

```
print("Contando del 2 al 6:")
for numero in range(2, 7): # Genera números desde el 2 hasta el 6
    print(numero)
```

Salida:

```
Contando del 2 al 6:
2
3
4
5
6
```

c) `range(start, stop, step)`: Definiendo el tamaño del paso.

También puedes indicar de cuánto en cuánto quieres que avance el contador.

Ejemplo: Imprimir los números pares del 0 al 10.

```
print("Números pares del 0 al 10:")
for numero in range(0, 11, 2): # Va de 2 en 2
    print(numero)
```

Salida:

```
Números pares del 0 al 10:
0
2
4
6
8
10
```

2. Recorriendo una Lista

Una de las utilidades más potentes del `for` es recorrer los elementos de una lista. La variable temporal tomará el valor de cada elemento de la lista, en orden.

Ejemplo: Saludar a una lista de amigos.

```
amigos = ["Ana", "Juan", "Carlos", "Sofía"]

print("Mis amigos:")
for nombre in amigos:
    print(f"¡Hola, {nombre}!")
```

Salida:

```
Mis amigos:
¡Hola, Ana!
¡Hola, Juan!
¡Hola, Carlos!
¡Hola, Sofía!
```

3. Recorriendo un String (Cadena de Texto)

Un string es, en esencia, una secuencia de caracteres. Por lo tanto, también podemos recorrerlo con un bucle `for`.

Ejemplo: Deletrear una palabra.

```
palabra = "Python"

print(f"Deletreando la palabra '{palabra}':")
for letra in palabra:
    print(letra)
```

Salida:

```
Deletreando la palabra 'Python':
P
y
t
h
o
n
```

4. Usando un Contador junto con la Secuencia: `enumerate()`

A veces, no solo quieres el elemento, sino también su posición (su índice). Por ejemplo, en una lista `["a", "b", "c"]`, el elemento `"a"` está en la posición 0, `"b"` en la 1, y así sucesivamente.

La función `enumerate()` nos devuelve una tupla con el índice y el valor en cada iteración.

Ejemplo: Mostrar una lista de compras con su numeración.

```
lista_de_compras = ["Manzanas", "Pan", "Leche", "Huevos"]

print("Lista de compras:")
for indice, producto in enumerate(lista_de_compras):
    print(f"{indice + 1}. {producto}") # Sumamos 1 al índice para que no empiece en 0
```

Salida:

```
Lista de compras:
1. Manzanas
2. Pan
3. Leche
4. Huevos
```

En este caso, en la primera iteración, `indice` es `0` y `producto` es `"Manzanas"`. En la segunda, `indice` es `1` y `producto` es `"Pan"`, y así sucesivamente.

Resumen

Forma de Uso	Sintaxis de Ejemplo	¿Para qué sirve?
Con `range()`	`for i in range(10):`	Para ejecutar un bloque de código un número específico de veces. Ideal para contadores.
Con una Lista	`for item in mi_lista:`	Para realizar una operación en cada uno de los elementos de una lista.
Con un String	`for caracter in mi_texto:`	Para procesar cada caracter de una cadena de texto de forma individual.
Con `enumerate()`	`for i, valor in enumerate(mi_lista):`	Cuando necesitas tanto el elemento como su índice (posición) dentro de la secuencia.