El futuro digital es de todos

MinTIC

Ejercicios con for

Rogerio Orlando Beltrán Castro

Universidad Industrial de Santander

MisiónTIC2022

El bucle for

En general, un bucle es una estructura de control que repite un bloque de instrucciones. Un bucle for es un bucle que repite el bloque de instrucciones un número prederminado de veces. El bloque de instrucciones que se repite se suele llamar cuerpo del bucle y cada repetición se suele llamar iteración.

La sintaxis de un bucle for es la siguiente:

```
for variable in elemento iterable (lista, cadena, range, etc.):
    cuerpo del bucle
```

No es necesario definir la variable de control antes del bucle, aunque se puede utilizar como variable de control una variable ya definida en el programa.

El cuerpo del bucle se ejecuta tantas veces como elementos tenga el elemento recorrible (elementos de una lista o de un range(), caracteres de una cadena, etc.)

Ejemplo de bucle 1

```
print("Comienzo")
for i in [0, 1, 2]:
    print("Hola",i)
print()
print("Final")
```

```
Comienzo
Hola 0
Hola 1
Hola 5

Final
```

Los valores que toma la variable no son importantes, lo que importa es que la lista tiene tres elementos y por tanto el bucle se ejecuta tres veces.

Ejemplo de bucle 2

```
print("Comienzo")
for i in []:
    print("Hola ")
print()
print("Final")
```

```
Comienzo

Final
```

Si la lista está vacía, el bucle no se ejecuta.

Ejemplo de bucle 3

```
print("Comienzo")
for i in [1, 1, 1]:
    print("Hola ",i)
print()
print("Final")
```

```
Comienzo
Hola  1
Hola  1
Hola  1

Final
```

Los valores que toma la variable no son importantes, lo que importa es que la lista tiene tres elementos y por tanto el bucle se ejecuta tres veces.

Ejemplo de bucle 4

```
print("Comienzo")
for _ in [0, 1, 2]:
    print("Hola ")
print()
print("Final")
```

```
Comienzo
Hola
Hola
Hola

Final
```

Si la variable de control no se va a utilizar en el cuerpo del bucle, como en los ejemplos anteriores, se puede utilizar el guion (_) en vez de un nombre de variable. Esta notación no tiene ninguna consecuencia con respecto al funcionamiento del programa, pero sirve de ayuda a la persona que esté leyendo el código fuente, que sabe así que los valores no se van a utilizar.

Ejemplo de bucle 5

```
print("Comienzo")
for i in [3, 4, 5]:
    print(f"Hola. Ahora i vale {i} y su cuadrado {i ** 2}")
print("Final")
```

```
Comienzo
Hola. Ahora i vale 3 y su cuadrado 9
Hola. Ahora i vale 4 y su cuadrado 16
Hola. Ahora i vale 5 y su cuadrado 25
Final
```

En los ejemplos anteriores, la variable de control "i" no se utilizaba en el bloque de instrucciones, pero en muchos casos sí que se utiliza. Cuando se utiliza, hay que tener en cuenta que la variable de control va tomando los valores del elemento recorrible

Ejemplo de bucle 6

```
print("Comienzo")
for i in ["Alba", "Benito", 27]:
    print(f"Hola. Ahora i vale {i}")
print("Final")
```

```
Comienzo
Hola. Ahora i vale Alba
Hola. Ahora i vale Benito
Hola. Ahora i vale 27
Final
```

La lista puede contener cualquier tipo de elementos, no sólo números. El bucle se repetirá siempre tantas veces como elementos tenga la lista y la variable irá tomando los valores de uno en uno

Ejemplo de bucle 7

```
print("Comienzo")
for numero in  [0, 1, 2, 3]:
    print(f"{numero} * {numero} = {numero ** 2}")
print("Final")
```

Comienzo

0 * 0 = 0

1 * 1 = 1

2 * 2 = 4

3 * 3 = 9

Final

La costumbre más extendida es utilizar la letra *i* como nombre de la variable de control, pero se puede utilizar cualquier otro nombre válido

Ejemplo de bucle 8

```
i = 10

print(f"El bucle no ha comenzado. Ahora i vale {i}")


for i in [0, 1, 2, 3, 4]:
    print(f"{i} * {i} = {i ** 2}")


print(f"El bucle ha terminado. Ahora i vale {i}")
```

El bucle no ha comenzado. Ahora i vale 10

0 * 0 = 0

1 * 1 = 1

2 * 2 = 4

3 * 3 = 9

4 * 4 = 16

El bucle ha terminado. Ahora i vale 4

La variable de control puede ser una variable empleada antes del bucle. El valor que tuviera la variable no afecta a la ejecución del bucle, pero cuando termina el bucle, la variable de control conserva el último valor asignado

Ejemplo de bucle 9

```
for i in [0, 1, 2]:
    print(f"{i} * {i} = {i ** 2}")

print()

for i in [0, 1, 2, 3]:
    print(f"{i} * {i} * {i} = {i ** 3}")
```

```
0 * 0 = 0
1 * 1 = 1
2 * 2 = 4

0 * 0 * 0 = 0
1 * 1 * 1 = 1
2 * 2 * 2 = 8
3 * 3 * 3 = 27
```

Cuando se escriben dos o más bucles seguidos, la costumbre es utilizar el mismo nombre de variable puesto que cada bucle establece los valores de la variable sin importar los valores anteriores

Ejemplo de bucle 10

```
for i in "AMIGO":
    print(f"Dame una {i}")
print("¡AMIGO!")
```

```
Dame una A
Dame una M
Dame una I
Dame una G
Dame una O
¡AMIGO!
```

En vez de una lista se puede escribir una cadena, en cuyo caso la variable de control va tomando como valor cada uno de los caracteres

Ejemplo de bucle 11

```
print("Comienzo")
for i in range(3):
    print("Hola ", end="")
print()
print("Final")
```

```
Comienzo
Hola Hola Hola
Final
```

En los ejemplos anteriores se ha utilizado una lista para facilitar la comprensión del funcionamiento de los bucles pero, si es posible hacerlo, se recomienda utilizar tipos range(), entre otros motivos porque durante la ejecución del programa ocupan menos memoria en el ordenador.

Ejemplo de bucle 12

```
print("Comienzo")
for i in range(10):
    print("Hola ", end="")
print()
print("Final")
```

```
Comienzo
Hola Hola Hola Hola Hola Hola
Hola Hola Hola Hola
Final
```

Otra de las ventajas de utilizar tipos range() es que el argumento del tipo range() controla el número de veces que se ejecuta el bucle.
En el ejemplo anterior basta cambiar el argumento para que el programa salude muchas más veces.

Ejemplo de bucle 13

```
print("Comienzo")
for i in range(10):
    print("Hola ", end="")
print()
print("Final")
```

```
¿Cuántas veces quiere que le salude? 5
Hola Hola Hola Hola Hola
Adiós
```

Esto permite que el número de iteraciones dependa del desarrollo del programa. En el ejemplo siguiente es el usuario quien decide cuántas veces se ejecuta el bucle.

Bucles anidados

Se habla de bucles anidados cuando un bucle se encuentra en el bloque de instrucciones de otro bloque.

Al bucle que se encuentra dentro del otro se le puede denominar bucle interior o bucle interno. El otro bucle sería el bucle exterior o bucle externo.

Los bucles pueden tener cualquier nivel de anidamiento (un bucle dentro de otro bucle dentro de un tercero, etc.).

Aunque en Python no es necesario, se recomienda que los nombres de las variables de control de los bucles anidados no coincidan, para evitar ambigüedades.

Ejemplo de bucle 1

```
for i in [0, 1, 2]:
    for j in [0, 1]:
        print(f"i vale {i} y j vale {j}")
```

```
i vale 0 y j vale 0
i vale 0 y j vale 1
i vale 1 y j vale 0
i vale 1 y j vale 1
i vale 2 y j vale 0
i vale 2 y j vale 1
```

Se dice que las variables de los bucles son **independientes** cuando los valores que toma la variable de control del bucle interno **no** dependen del valor de la variable de control del bucle externo. En el ejemplo anterior, el bucle externo (el controlado por i) se ejecuta 3 veces y el bucle interno (el controlado por j) se ejecuta dos veces por cada valor de i. Por ello la instrucción print() se ejecuta en total 6 veces (3 veces que se ejecuta el bucle externo x 2 veces que se ejecuta cada vez el bucle interno = 6 veces).
En general, el número de veces que se ejecuta el bloque de instrucciones del bucle interno es el producto de las veces que se ejecuta cada bucle.

Ejemplo de bucle 2

```
for i in range(3):
    print(f"i (externa) vale {i}")
    for i in range(2):
        print(f"i (interna) vale {i}")
```

```
i (externa) vale 0
i (interna) vale 0
i (interna) vale 1
i (externa) vale 1
i (interna) vale 0
i (interna) vale 1
i (externa) vale 2
i (interna) vale 0
i (interna) vale 1
```

La costumbre más extendida es utilizar la letra "i" como nombre de la variable de control del bucle externo y la letra "j" como nombre de la variable de control del bucle interno (o "k" si hay un tercer nivel de anidamiento), pero se puede utilizar cualquier otro nombre válido.
En Python se puede incluso utilizar la misma variable en los dos bucles anidados porque Python las trata como si fueran dos variables distintas

Ejemplo de bucle 3

```
for i in [1, 2, 3]:
    for j in range(i):
        print(f"i vale {i} y j vale {j}")
```

```
i vale 1 y j vale 0
i vale 2 y j vale 0
i vale 2 y j vale 1
i vale 3 y j vale 0
i vale 3 y j vale 1
i vale 3 y j vale 2
```

Se dice que las variables de los bucles son **dependientes** cuando los valores que toma la variable de control del bucle interno dependen del valor de la variable de control del bucle externo