P04 El bucle for en Python

1.- El bucle for y la instrucción range

En las lecciones anteriores hemos aprendido a emplear programas secuenciales y condicionales. Pero en muchas ocasiones necesitaríamos repetir parte del código varias veces. Para esto emplearemos bucle.

En Python hay do tipos de bucle, `for` y `while` , en este tema estudiaremos el bucle `for`

En primer lugar probemos a realizar un programa muy sencillo

```
for character in 'hello':
    print(character)
```

A veces no querremos repetir el código para las letras de una palabra, sino que querremos usar números consecutivos. Para ello usaremos la siguiente estructura

```
for i in range(5, 8):
    print(i, i ** 2)
print('fin del bucle')
# 5 25
# 6 36
# 7 49
# fin del bucle
```

La función `range(min_value, max_value)` genera una secuencia con los números `min_value`, `min_value + 1`, ..., `max_value - 1`. El último numero no esta incluido en ella.

Hay una forma reducida de range() `range(max_value)`, En este caso min_value se entiende implícitamente como 0

```
for i in range(3):
    print(i)
# 0
# 1
# 2
```

Empleando esta estructura podemos repetir varias veces la misma acción:

```
for i in range(2 ** 2):
    print('Hello, world!')
```

Al igual que con las estructuras if-else la indentación indica que instrucciones forman parte del bloque del for.

Veamos un ejemplo más complejo en el que empleamos for para sumar todos los números desde 1 a 5

```
resultado = 0
n = 5
for i in range(1, n + 1):
    resultado += i
    # esto    ^^ es una forma abreviada de escribir
    # resultado = resultado + i
print(resultado)
```

Es importante notar que se emplea como valor máximo en range se usa `n + 1` para poder hacer `i` igual a n en el último paso.

Para iterar con un valor decreciente podemos añadir un parámetro a la instrucción range, llamándola con tres argumentos. Cuando se omite, step se entiende de forma implícita como 1, aunque puede tomar cualquier valor distinto de cero. Siempre incluye start_value y excluye end_value.

```
for i in range(10, 0, -2):
    print(i)
# 10
# 8
# 6
# 4
# 2
```

2.- Sintaxis.

2.1.- Bucle for

for variable in secuencia:

    bloque del for

Es importante señalar que siempre necesitaremos una variable, aunque no la vayamos a emplear para nada. Esta variable se suele denominar contador, tradicionalmente cuando se itera sobre números enteros se emplean los identificadores i,j,k

En general la secuencia vendrá definida por una lista, volveremos más adelante sobre ellas, por ahora nos quedaremos en que podemos iterar sobre los caracteres de una variable tipo string o usar la instrucción range, o también podremos construirla nosotros

```
 i = 1
for color in 'rojo', 'naranja', 'amarillo', 'verde', 'cyan', 'blue', 'violet':
    print('#', i, 'color del arco iris es', color)
    i = i + 1
```

Cuando la construyamos nosotros no es necesario que todos los elementos sean del mismo tipo

```
for i in 1, 2, 3, 'one', 'two', 'three':
    print(i)
```

2.2.- Range

range(start_value,end_value,step)

Devuelve una lista con los valores que van de start_value hasta end_value (sin incluirlo) con incrementos de step.

Si se llama con un único parámetro range(x) lo entenderá como range(0,x,1)

Si se llama con dos parámetros range(x,y) lo entenderá como range(x,y,1)

Si se llama con parámetros que no tengan sentido (Ej range(2,1)) no devolverá ningún valor, pero no dará ningún error, y el bloque for no se ejecutará

3.- Mejorando la función print().

Por defecto la función print() imprime todos sus argumentos separándolos con un espacio y poniendo un salto de línea al terminar. Esto se puede cambiar usando los argumentos sep y end

Comprobar los efectos con los siguientes ejemplos.

```
print(1, 2, 3)
print(4, 5, 6)
print(1, 2, 3, sep=', ', end='. ')
print(4, 5, 6, sep=', ', end='. ')
print()
print(1, 2, 3, sep='', end=' -- ')
print(4, 5, 6, sep=' * ', end='.')
```

Ejercicios:

P1 Series 1
El programa debe preguntar dos enteros A, B e imprimir todos los enteros de A a B en orden ascendente (ambos incluidos).
Nótese que no tienen por que darse de menor a mayor asi que habrá que hacer uso de if.

P2 Series 2
El programa debe preguntar dos enteros A,B e imprimir todos los enteros de A a B (ambos incluidos) en orden ascendente si A<B o en orden descendente si A>B.

P3 Suma de 10 números
El programa debe preguntar 10 números y dar su suma. Usar el menor número posible de variables.
Pista: El programa debería poder adaptarse para realizar la suma de 50 números cambiando solamente una línea.

P4 Suma de N números
El programa debe preguntar un entero N, y después preguntar N números y devolver su suma.
Debería realizarse fácilmente a partir del P3

P5 Suma de cubos.
Dado un número N (que se debe preguntar) calcular la siguiente suma.

$$1^3 + 2^3 + \cdots + N^3$$

P6 Factorial
Dado un número N (que se debe preguntar) calcular su factorial

P7 Número de ceros
El programa debe preguntar un entero N, y después preguntar N números y devolver el número de veces que se ha introducido el número cero.

P8 Sumar factoriales. (Difícil)

Dado un número N (que se debe preguntar) calcular

$$1! + 2! + \cdots + N!$$

Se debería realizar con un único bucle

P9 Escalera
Para un entero dado n<=9 imprimir una escalera de n pasos. El paso número i consiste en los enteros del 1 al i sin espacios entre ellos.
Es necesario usar los nuevos parámetros de la función print().