

## Ciclos: Ejemplo “Sumatoria”

Tenemos la función `sumatoria` especificada e implementada de la siguiente manera:

```
1  def sumatoria(n:int) -> int:
2      ''' Requiere: n>=0
3          Devuelve: la suma de los números enteros entre 0 y n (inclusive). '''
4      res:int = 0
5      i:int = 0
6      # (A)
7      while i < n:
8          # (B)
9          i = i + 1
10         res = res + i
11         # (C)
12     # (D)
13     return res
```

Se pide:

1. Mostrar que el ciclo termina.
2. Elegir un predicado invariante  $\mathcal{I}$ , y usarlo para mostrar que cuando termina el ciclo, la función devuelve el valor indicado en la especificación.

### 1. Terminación del ciclo

- Antes del ciclo, la variable `i` se inicializa en 0 (línea 5).
- En cada ejecución del cuerpo del ciclo, `i` se incrementa en 1 (línea 9).
- Por la cláusula *Requiere* sabemos que la variable entera `n` es mayor o igual a cero y, además, no se modifica en el cuerpo del ciclo.
- Entonces, es inevitable que en algún momento, `i` llegue al valor de `n`.
- En ese momento, la condición `i < n` será `False`, por lo que el ciclo terminará.  $\square$

### 2. Correctitud del ciclo

Primero identifiquemos qué cosas podemos afirmar sobre las variables del programa, que sean verdaderas en los puntos (A), (B), (C) y (D) indicados en el código de arriba.

Para entender mejor el código, ejecutemos manualmente esta función para un caso particular, `n=3`:

1. Tanto `i` como `res` empiezan valiendo 0.
2. La condición `i < n` es verdadera ( $0 < 3$ ), por lo que ingresamos al cuerpo del ciclo.
3. Al terminar la primera iteración del ciclo, `i` vale 1 y `res` también vale 1.
4. La condición `i < n` es verdadera ( $1 < 3$ ), por lo que ingresamos al cuerpo del ciclo.
5. Al terminar la segunda iteración del ciclo, `i` vale 2 y `res` vale 3.
6. La condición `i < n` es verdadera ( $2 < 3$ ), por lo que ingresamos al cuerpo del ciclo.
7. Al terminar la tercera iteración del ciclo, `i` vale 3 y `res` vale 6.
8. Ahora la condición `i < n` es **falsa** (no es cierto que  $3 < 3$ ), por lo que esta vez **no** ingresamos al cuerpo del ciclo.
9. Se retorna el valor actual de `res`, que es 6.

Pasemos a una tabla los valores de las variables **i** y **res**:

<b>i</b>	<b>res</b>
0	0 = 0
1	1 = 0 + 1
2	3 = 0 + 1 + 2
3	6 = 0 + 1 + 2 + 3

Viendo esta tabla, podemos afirmar dos cosas que son verdaderas en cada iteración:

- La variable **i** se encuentra entre 0 y **n** inclusive (recordemos que **n**=3). Es decir:  $0 \leq i \leq n$ .
- La variable **res** vale la suma de los enteros entre 0 e **i** (inclusive). Es decir:  $\text{res} = \sum_{j=0}^i j$ .

La notación matemática  $\sum_{j=n}^m j$  es una abreviatura de  $n + (n + 1) + (n + 2) + \dots + m$ .  
 Por ejemplo,  $\sum_{j=4}^7 j = 4 + 5 + 6 + 7 = 22$ .  
 Casos particulares: cuando los límites inferior y superior son iguales,  $\sum_{j=n}^n j$  es igual a **n**;  
 cuando  $n > m$ ,  $\sum_{j=n}^m j$  es igual a cero.  
 $\Sigma$  es la letra griega *sigma* en mayúscula, siendo  $\sigma$  su minúscula. ¡No es una “E griega”!

Así, nuestro predicado invariante  $\mathcal{I}$  será:  **$0 \leq i \leq n$ , y **res** vale la suma de los enteros entre 0 e **i**.**

Notar que proponemos este  $\mathcal{I}$  para todo valor posible de **n**; ya no para el caso particular **n**=3, que habíamos usado para intentar entender el comportamiento del código.

Sigamos ahora el siguiente razonamiento, que nos lleva desde el principio hasta el final de la ejecución de la función *sumatoria*:

- $\mathcal{I}$  es verdadero en el punto (A), donde **i** vale 0 y **res** también vale 0.
- Supongamos que  $\mathcal{I}$  es verdadero en el punto (B). Si llamamos  $\phi$  al valor que tiene **i** en ese momento, las siguientes afirmaciones son verdaderas:

$$i = \phi$$

$$\text{res} = \sum_{j=0}^i j = \sum_{j=0}^{\phi} j$$

En el cuerpo del ciclo, **i** se incrementa en 1 (línea 9), y a **res** se le suma un término con el nuevo valor de **i** (línea 10). En consecuencia, al terminar el cuerpo del ciclo, en el punto (C), valen las siguientes afirmaciones:

$$i = \phi + 1$$

$$\text{res} = \sum_{j=0}^{\phi} j + (\phi + 1) = \sum_{j=0}^{\phi+1} j = \sum_{j=0}^i j$$

En otras palabras,  $\mathcal{I}$  vuelve a ser verdadero al terminar el cuerpo del ciclo, en el punto (C).

- De esta manera, sabemos que en cada iteración del ciclo,  $\mathcal{I}$  empieza y termina siendo verdadero. Después de (C) se evalúa la condición  $i < n$ , lo cual no modifica el valor de ninguna variable. Así, mientras la condición resulte verdadera, a lo largo de sucesivas iteraciones tendremos que  $\mathcal{I}$  vale en (B), vale en (C), vale en (B), vale en (C), vale en (B), vale en (C), ...
- En algún momento el ciclo termina porque la condición resulta falsa. Dado que evaluar la condición  $i < n$  no modifica el valor de las variables,  $\mathcal{I}$  sigue siendo verdadero al llegar al punto (D), y es fácil ver que **i** vale **n**. Entonces, en este momento estamos en condiciones de afirmar que **res vale la suma de los enteros entre 0 y n**. Es decir:  $\text{res} = \sum_{j=0}^n j$ . Y eso es precisamente lo que la función *sumatoria* debe devolver de acuerdo a su especificación.  $\square$