

Estructuras de control y ciclos

Estructura For

Aldo Sayeg Pasos Trejo

Física Computacional
Facultad de Ciencias
Universidad Nacional Autónoma de México

1 de octubre de 2020

Estructura for

Parece que la estructura `while` puede ser un poco peligrosa pues se presta a que el ciclo nunca pueda concluir. ¿Podríamos tener una estructura que también ejecute código de manera repetitiva pero no mientras se cumpla una condición si no un número finito de veces?

Esa estructura existe, y se le llama normalmente **ciclo for**. Es una estructura más compleja pues la manera de especificarle las veces que queremos que repita código (es decir, las **iteraciones**) cambia radicalmente entre lenguajes.

Programas con for

Programa 1: for para imprimir todos los números de 1 a 100

Input :

Output:

```
1 for  $i = 1, 2, 3, \dots 100$  do  
2   |   Imprimir  $i$   
3 end
```

Programas con for

Programa 2: for para imprimir todos los múltiplos de 3
menores a 100

Input :

Output:

```
1 for  $i = 1, 2, 3, \dots 100$  do  
2   |   if  $i \% 3 = 0$  then  
3   |       |   Imprimir  $i$   
4   |   end  
5 end
```

Programas con for

Programa 3: for para obtener $\sum_{i=1}^{100} i$

Input :

Output:

```
1 Sea suma = 0
2 for i = 1, 2, 3, ... 100 do
3   |   Sea suma = suma + i
4 end
5 Imprimir suma
```

Programas con for

Programa 4: for para calcular $\sum_{i=1}^{100} \sqrt{(i+60)^3}$

Input :

Output:

```
1 Sea suma = 0
2 for i = 1, 2, 3, ... 100 do
3   | Sea suma = suma +  $\sqrt{(i+60)^3}$ 
4 end
5 Imprimir suma
```

Programas con for

Programa 5: for como en Python y Julia

Input :

Output:

```
1 Sea  $suma = 0$   
2 Sea  $A = [1, 2, 3, \dots, 100]$   
3 for  $i \in A$  do  
4   | Sea  $suma = suma + \sqrt{(i + 60)^3}$   
5 end  
6 Imprimir  $suma$ 
```

Programas con for

Programa 6: for como en Python y Julia

Input :

Output:

```
1 Sea  $F = ["melon", "sandia", "papaya"]$   
2 for  $fruta \in F$  do  
3   |   Imprimir  $fruta$   
4 end
```

Teorema de la programación estructurada

Valdría la pena preguntarse si existe alguna otra estructura de control que nos pueda ser útil o que nos haga falta para que nuestro lenguaje de alto nivel pueda hacer cualquier procedimiento algorítmico, es decir, ser Turing-completo.

Sea ha demostrado (1966, Böhn y Jacopini) que los ciclos `if`, `while` y `for` son suficientes para realizar cualquier cómputo [1]. Eso es llamado el **teorema de la computación estructurada**

En la práctica, algunos lenguajes cuentan con otras estructuras o modificaciones a las ya planteadas que logran darnos ventajas, pero son equivalentes.

Referencias



Subrata Dasgupta.

Computer Science: a very short introduction, volume 466.

Oxford University Press, 2016.