

# Pseudocódigo

El **pseudocódigo** consiste en expresar mediante un lenguaje coloquial las operaciones que describen el algoritmo asociado a un programa informático. Básicamente, se trata de traducir las operaciones que se describen en un organigrama a frases del lenguaje común, sin emplear símbolos.

El lenguaje utilizado para describir los algoritmos mediante pseudocódigo, debe ser fácil de interpretar por parte de una persona. Las palabras y expresiones utilizadas en los mismos deberán indicar de forma clara y precisa las tareas a realizar.

Y es que, el objetivo del pseudocódigo es intentar describir un **algoritmo** de la forma más detallada posible y, a su vez, próxima a un lenguaje de programación de alto nivel, de manera que la traducción final al código real durante la fase de implementación sea lo más sencilla posible, de ahí que se le llame pseudocódigo.

Seguidamente vamos a ver unos ejemplos que nos van a ir aclarando como plantear un pseudocódigo y las estructuras típicas que se utilizan.

En este primer listado te presentamos la versión en pseudocódigo del primer ejemplo que realizamos con ordinogramas, correspondiente al algoritmo para **leer dos números y mostrar el mayor de los dos**:

```
Inicio
    leer a, b
    si (a==b) entonces
        mostrar "Son iguales"
    sino
    si (a>b) entonces
        mostrar "El mayor es", a
    sino
        mostrar "El mayor es", b
    fin si
fin si
Fin
```

En el caso de estructuras complejas, como las instrucciones alternativas, utilizamos palabras especiales como **si..entonces..sino..fin si**, para definir las operaciones y

delimitar su ámbito.

Durante la definición de un pseudocódigo es conveniente indentar (escribir con espacios o tabulaciones a la izquierda) aquellas instrucciones o bloques de instrucciones que forman parte de otras, a fin de aclarar la estructura del algoritmo.

Aunque hay lenguajes de programación de alto nivel que si lo hacen, nosotros no haremos distinción entre mayúsculas y minúsculas a la hora de escribir un pseudocódigo.

En este otro ejemplo tenemos el bloque de pseudocódigo que describe el algoritmo a seguir por un programa encargado de **mostrar la suma de todos los números naturales comprendidos entre 1 y un número leído**:

```
Inicio
suma=0
cont=1
leer n
    etiquet1:
    suma=suma+cont
    si(cont<n) entonces
        cont=cont+1
        ir a etiquet1
    sino
        mostrar suma
    fin si
Fin
```

En el pseudocódigo anterior vemos el uso de etiquetas, como la expresión “etiqueta 1:”, que sirven para marcar ciertas partes del programa, de modo que podamos utilizar dichas etiquetas como referencia a la hora pasar el control del programa a ese punto.

Sin la utilización de **saltos**, que muchas veces dificultan el seguimiento de un algoritmo, el programa anterior podría haberse escrito de esta otra manera utilizando una estructura repetitiva:

```
Inicio
suma=0
cont=1
leer n
```

```
mientras (cont<=n)
    suma=suma+cont
    cont=cont+1
fin mientras
mostrar suma
Fin
```

Con la estructura repetitiva **mientras (while)**, lo que queremos expresar es que mientras se cumpla la condición indicada en el paréntesis, el programa tendrá que ejecutar el bloque de sentencias codificadas en su interior (las que aparecen entre mientras y fin mientras). Su funcionamiento es el siguiente: cuando el programa llega a la instrucción mientras, se comprueba la condición y, si esta es cierta, se ejecutará el bloque de sentencias. Después de ejecutar la última sentencia del bloque, el programa vuelve a comprobar la condición del mientras y, si vuelve a cumplirse, de nuevo se ejecuta el conjunto de instrucciones.

Así sucesivamente hasta que la **condición** sea falsa, en cuyo caso el programa continuará con las instrucciones situadas después de fin mientras.

Existen distintas variantes a la hora de implementar una **estructura repetitiva**, como que la condición sea evaluada después de la ejecución de las instrucciones en vez de antes, o que el bloque de sentencias se ejecute un número definido de veces controlado por un contador.

El siguiente pseudocódigo corresponde a un algoritmo que realiza la **lectura y suma de números hasta que el usuario indica que no quiere introducir más números**:

```
Inicio
suma=0

Hacer
    leer n
    suma=suma+n
    mostrar "¿Desea introducir otro número?"
    leer op
Mientras (op=="si")

Mostrar suma
Fin
```

En este caso utilizamos la expresión Hacer...Mientras (**do while**), para ejecutar primero el bloque de sentencias que se deben repetir y preguntas al final por la condición, de modo que si esta condición se cumple, se volverá a ejecutar de nuevo el conjunto de instrucciones indicadas dentro de hacer.

## 1.1. Normas en la creación de pseudocódigo

Aunque cada persona es libre de definir sus propias normas a la hora de crear pseudocódigo, a fin de conseguir este objetivo, es necesario seguir una serie de reglas básicas y de sentido común, entre las que podríamos destacar:

- Disponer de un juego limitado de instrucciones. Es importante que a la hora de definir un algoritmo mediante pseudocódigo, utilicemos siempre el mismo juego de instrucciones y con ellas resolver cualquier problema de programación.
- Utilizar estructuras lógicas de control. Además de lo que serían las instrucciones de proceso (asignación de datos a una variable, operación aritmética entre variables, etc.), se debe disponer de un juego de instrucciones que expresen las operaciones de control de flujo que se llevan a cabo habitualmente en cualquier programa, como las alternativas simples y múltiples o las de tipo repetitivo.
- Separación de datos y código. Cuando se van a manejar varios datos en un programa, conviene separar la declaración de esos datos a utilizar de lo que serían las instrucciones de manipulación de los mismos.

Según la última de las reglas que acabamos de presentar, conviene a la hora de diseñar el pseudocódigo de un algoritmo definir previamente las variables de los datos que se van a manipular y, cuando proceda, inicializar las mismas. Esto permite, antes de empezar con el código, aclarar con qué datos vamos a trabajar.

A la hora de declarar las variables, se indicará el nombre de la variable seguido del tipo de dato con el que vamos a trabajar:

```
nombre_variable tipo
```

A continuación otro algoritmo de ejemplo en pseudocódigo. Corresponde al de un programa encargado de leer un número y mostrar el factorial de dicho número.

El factorial de un número se calcula multiplicando todos los números naturales menores de ese número hasta 1. Por ejemplo, el factorial del 5 se calcularía:

5\*4\*3\*2\*1

Pseudocódigo:

```
Inicio
Datos:
    factorial entero
    cont entero
    n entero
Código:
    factorial=1
    Leer n
    cont=n
    Mientras (cont>=1)
        factorial=factorial*cont
        cont=cont-1
    Fin mientras
    mostrar factorial
Fin
```

En este ejemplo mostrado se trata de ir multiplicando los números naturales comprendidos entre 1 y el número leído.

Una vez más, para ir ejecutando repetidas veces una tarea tenemos que echar mano de la instrucción mientras.

Para controlar el número de multiplicaciones realizadas, utilizamos una variable contador que se inicializa al número leído y que con cada multiplicación se decrementa en una unidad. La variable factorial se utiliza como acumulador de las multiplicaciones.