

CARRERA PROFESIONAL

# **DESARROLLO DE SISTEMAS DE INFORMACIÓN FUNDAMENTOS DE PROGRAMACIÓN**

Tema

**ESTRUCTURA DE CONTROL MULTIPLE**

### Conceptos Basicos

Los programas utilizados hasta este momento han examinado conceptos de programación, tales como entradas, salidas, asignaciones, expresiones y operaciones, sentencias secuenciales y de selección. Sin embargo, muchos problemas requieren de características de repetición, en las que algunos cálculos o secuencia de instrucciones se repiten una y otra vez, utilizando diferentes conjuntos de datos. Ejemplos de tales tareas repetitivas incluyen verificaciones (chequeos) de entradas de datos de usuarios hasta que se introduce una entrada aceptable, tal como una contraseña válida; conteo y acumulación de totales parciales; aceptación constante de entradas de datos y recálculos de valores de salida, cuyo proceso sólo se detiene cuando se introduce o se presenta un valor centinela.

Este capítulo examina los diferentes métodos que utilizan los programadores para construir secciones de código repetitivas. Se describe y analiza el concepto de bucle como la sección de código que se repite y que se denomina así ya que cuando termina la ejecución de la última sentencia el flujo de control vuelve a la primera sentencia y comienza otra repetición de las sentencias del código. Cada repetición se conoce como iteración o pasada a través del bucle.

Se estudian los bucles más típicos, tales como mientras, hacer-mientras, repetir-hasta que y desde (o para).

### Estructuras Repetitivas

Las computadoras están especialmente diseñadas para todas aquellas aplicaciones en las cuales una operación o conjunto de ellas deben repetirse muchas veces. Un tipo muy importante de estructura es el algoritmo necesario para repetir una o varias acciones un número determinado de veces. Un programa que lee una lista de números puede repetir

la misma secuencia de mensajes al usuario e instrucciones de lectura hasta que todos los números de un fichero se lean.

Las estructuras que repiten una secuencia de instrucciones un número determinado de veces se denominan bucles y se denomina iteración al hecho de repetir la ejecución de una secuencia de acciones. Un ejemplo aclarará la cuestión. Supongamos que se desea sumar una lista de números escritos desde teclado por ejemplo, calificaciones de los alumnos de una clase. El medio conocido hasta ahora es leer los números y añadir sus valores a una variable SUMA que contenga las sucesivas sumas parciales. La variable SUMA se hace igual a cero y a continuación se incrementa en

el valor del número cada vez que uno de ellos se lea. El algoritmo que resuelve este problema es:

```
algoritmo suma
var
    entero : SUMA, NUMERO
inicio
    SUMA ← 0
    leer(numero)
    SUMA ← SUMA + numero
    leer(numero)
    SUMA ← SUMA + numero
    leer(numero)
fin
```

y así sucesivamente para cada número de la lista. En otras palabras, el algoritmo repite muchas veces las acciones.

**leer(numero)**

SUMA  $\leftarrow$  SUMA + numero

Tales opciones repetidas se denominan bucles o lazos. La acción (o acciones) que se repite en un bucle se denomina iteración. Las dos principales preguntas a realizarse en el diseño de un bucle son ¿qué contiene el bucle? y ¿cuántas veces se debe repetir?

Cuando se utiliza un bucle para sumar una lista de números, se necesita saber cuántos números se han de sumar. Para ello necesitaremos conocer algún medio para detener el bucle. En el ejemplo anterior usaremos la técnica de solicitar al usuario el número que desea, por ejemplo, N. Existen dos procedimientos para contar el número de iteraciones, usar una variable TOTAL que se inicializa a la cantidad de números que se desea y a continuación se decrementa en uno cada vez que el bucle se repite (este procedimiento añade una acción más al cuerpo del bucle: TOTAL  $\leftarrow$  TOTAL - 1), o bien inicializar la variable TOTAL en 0 o en 1 e ir incrementando en uno a cada iteración hasta llegar al número deseado.

**algoritmo** suma\_numero

**var**

**entero** : N, TOTAL

**real** : NUMERO, SUMA

**inicio**

**leer(N)**

TOTAL  $\leftarrow$  N

SUMA  $\leftarrow$  0

**mientras** TOTAL > 0 **hacer**

**leer(NUMERO)**

SUMA  $\leftarrow$  SUMA + NUMERO

TOTAL ← TOTAL - 1

```

fin_mientras

escribir('La suma de los', N, 'números es', SUMA)

fin

```

El bucle podrá también haberse terminado poniendo cualquiera de estas condiciones:

- **hasta\_que** TOTAL sea cero
- **desde 1 hasta** N

Para detener la ejecución de los bucles se utiliza una condición de parada. El pseudocódigo de una estructura repetitiva tendrá siempre este formato:

```

inicio

    ///inicialización de variables

    repetir

        acciones S1, S2, ...

        salir según condición

        acciones Sn, Sn+1, ...

    fin_repetir

```

Aunque la condición de salida se indica en el formato anterior en el interior del bucle y existen lenguajes que así la contienen expresamente<sup>1</sup>, lo normal es que la condición se indique al final o al principio del bucle, y así se consideran tres tipos de instrucciones o estructuras repetitivas o iterativas generales y una particular que denominaremos iterar, que contiene la salida en el interior del bucle.

iterar	(loop)
mientras	(while)
hacer-mientras	(do-while)
repetir	(repeat)

desde

(for)



