

# Tema 8

## Metodología De Desarrollo De Programas (III)

### Operaciones abstractas

Los subprogramas permiten definir operaciones abstractas. Una abstracción es una visión simplificada de una cierta entidad, de la que sólo consideramos sus elementos esenciales, prescindiendo en lo posible de los detalles.

- **Especificación y realización:** Se definen dos posibles visiones:
  - **Abstracta:** es la especificación o interfaz de la operación, lo que hace.
  - **Detallada:** es la realización o implementación, el cómo lo hace.
- **Funciones. Argumentos:** En programación la idea de función surge al aplicar el concepto de abstracción a las expresiones aritméticas. Una expresión representa un nuevo valor obtenido por cálculo a partir de ciertos valores ya conocidos que se usan como operandos.
- **Acciones abstractas. Procedimientos:** De manera similar a como las funciones pueden ser consideradas como expresiones abstractas, parametrizadas, los procedimientos pueden ser considerados como acciones abstractas, igualmente parametrizadas. Un procedimiento representa una acción, que se define por separado, y que se invoca por su nombre.
 

**Procedimientos puros:** Se garantiza que un procedimiento cumple con esta cualidad si su realización no utiliza:

  - Variables externas al subprograma, a las que se accede directamente por su nombre, de acuerdo con las reglas de visibilidad de bloques.
  - Llamadas a otros subprogramas que no sean procedimientos o funciones puras.

### Desarrollo por refinamiento usando abstracciones

La metodología de programación estructurada puede ampliarse con la posibilidad de definir operaciones abstractas mediante subprogramas. Hay dos estrategias de desarrollo diferentes, según qué se escriba primero, si la definición de los subprogramas, o el programa principal que los utiliza.

- **Desarrollo descendente:** Es simplemente el desarrollo por refinamientos sucesivos, teniendo en cuenta además la posibilidad de definir operaciones abstractas. En cada etapa de refinamiento de una operación habrá que optar por una de las alternativas siguientes:
  - Considerar la operación como *operación terminal*, y CODIFICARLA mediante sentencias del lenguaje de programación.
  - Considerar la operación como *operación compleja*, y DESCOMPONERLA en otras más sencillas.
  - Considerar la operación como *operación abstracta*, y ESPECIFICARLA, escribiendo más adelante el subprograma que la realiza.
- **Reutilización:** La realización de ciertas operaciones como subprogramas independientes facilita lo que se llama *reutilización de software*. Si la operación identificada como operación abstracta tiene un cierto sentido en sí misma, es muy posible que resulte de utilidad en otros programas, además de en aquél para el cual se ha desarrollado. La escritura de otros programas que utilicen esa misma operación resulta más sencilla, ya que se aprovecha el código de su definición, que ya estaba escrito.
- **Desarrollo para reutilización:** Para aplicar de manera eficaz las técnicas de reutilización de software es preciso pensar en las posibles aplicaciones de un cierto subprograma en el momento de especificarlo, con independencia de las necesidades particulares del programa que se está desarrollando en ese momento.
- **Desarrollo ascendente:** Consiste en ir creando subprogramas que realicen operaciones significativas de utilidad para el programa que se intenta construir, hasta que finalmente sea posible escribir el programa principal, de manera relativamente sencilla, apoyándose en los subprogramas desarrollados hasta ese momento.

## ***Programas robustos***

---

Un programa se dice que es *robusto* si su operación se mantiene en condiciones controladas aunque se le suministren datos erróneos.

- **Programación a la defensiva:** Consiste en que cada programa o subprograma esté escrito de manera que desconfíe sistemáticamente de los datos o argumentos con que se invoca, y devuelva siempre como resultado:
  - El resultado correcto, si los datos son admisibles.
  - Una indicación precisa de error, si los datos no son admisibles.
- **Tratamiento de excepciones:** Ante la posibilidad de errores en los datos con que se opera, hay que considerar dos actividades diferentes:
  - Detección de la situación de error.
  - Corrección de la situación de error.