**Tema 2: Parametrización de acciones**

1. *Ventajas de la parametrización*

* Permite representar un conjunto potencialmente infinito de diferentes cálculos con un único texto de algoritmo que es una abstracción de ellos.
* Es un método para conseguir **generalidad**.
* **Definición de:**
* *Abstraer:* separar por medio de una operación intelectual las cualidades de un objeto para considerarlas aisladamente o para considerar el mismo objeto en su pura esencia o noción.
* *Generalidad:* mayoría, muchedumbre o casi totalidad de los individuos u objetos que componen una clase o todo sin determinación de persona o cosa particular.
* Facilitan una escritura de acciones sin referencias a variables del léxico que la incluye: INDEPENDENCIA DEL CONTEXTO EN EL QUE SE INTRODUCE LA ACCIÓN.

1. *Abstracción en acciones parametrizadas*

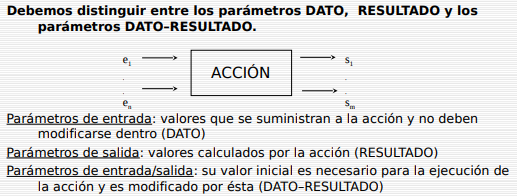
* **Doble abstracción:** A) Abstracción por especificación. B) Abstracción por parametrización.
* **Localidad:** Las acciones parametrizadas pueden ser escritas o comprendidas independientemente del contexto (léxico) en que se utilicen. Su uso implica comprender sólo su especificación
* **Facilitan:** la descomposición, legibilidad, mantenimiento, y reutilización

1. *Notación algorítmica para definir acciones*

* nombre: una acción (tp1 par1: td1;....tpn pan: tdn)
* donde: A) tpi: clase de parámetro (dato, resultado o dato–resultado ). B) pari: nombre (identificador) de parámetro. C) tdi: tipo de dato del parámetro. D) nombre: nombre de la acción
* Definir una nueva acción supone ENRIQUECER el conjunto de acciones primitivas o no primitivas disponibles
* Las acciones son la base para la DESCOMPOSICIÓN DE PROGRAMAS en el paradigma imperativo

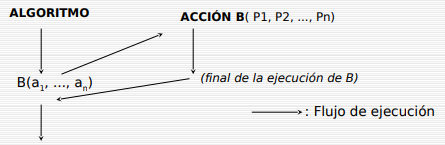
1. *Elección de los parámetros de una acción*

* Mayor parametrización ⇒ mayor independencia del contexto: más reutilizable
* Limitación de la parametrización, implica una adaptación mayor de la acción al contexto la reutilización directa es menos evidente



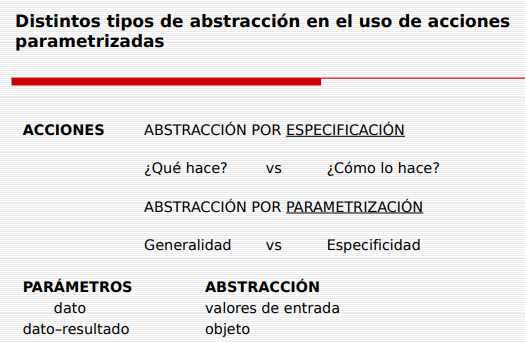
1. *Abstracción procedural en el uso de una acción*

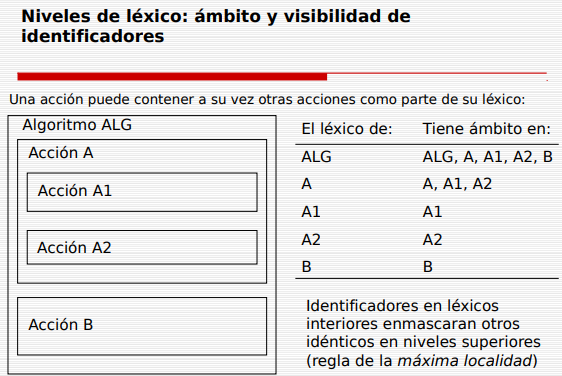
* Si el parámetro es un DATO de la acción: (en la llamada) parámetro ← argumento el argumento es una expresión del mismo tipo de dato que el parámetro.
* El parámetro es DATO y RESULTADO de la acción. (en la llamada) parámetro ← argumento (en el retorno) argumento ← parámetro el argumento es una variable del mismo tipo de dato que el parámetro.

****

1. *Consideraciones sobre el uso de parámetros*

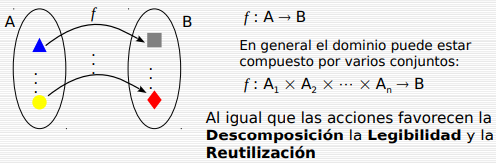
* Al emparejarse los parámetros de ambas listas (parámetros formales y argumentos) lo hacen de forma ordenada y deben coincidir en:
* Número de parámetros
* Tipo de parámetros
* Posición en las listas
* No es necesario que coincidan los identificadores de los argumentos y los parámetros formales:
* Los parámetros de una acción son parte de su léxico local
* No son visibles ni utilizables fuera de ella
* Sus identificadores pueden enmascarar elementos del léxico superior

****

****

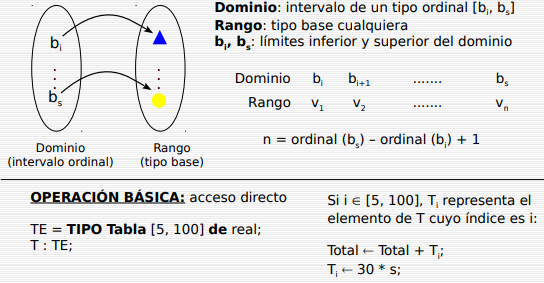
1. *Funciones*

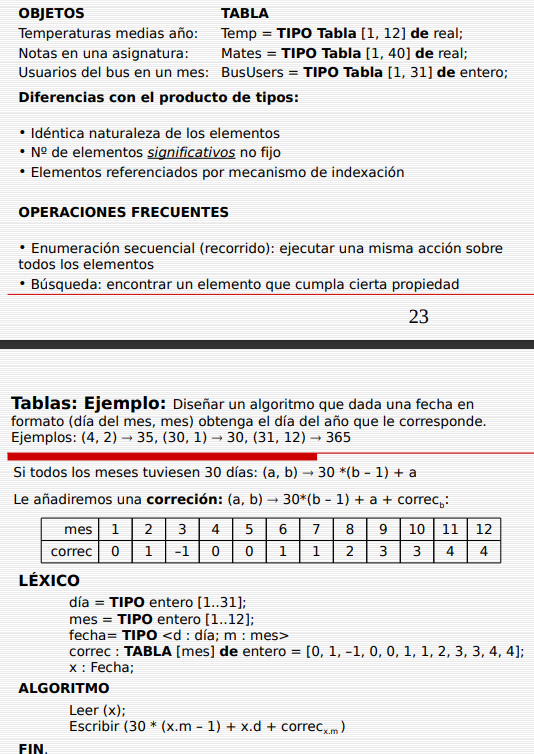
* Aunque se definen de forma parecida, tienen importantes diferencias con las acciones. En cuanto a su especificación:
* **Acciones**: Su efecto es modificar el estado del proceso, en términos de estado inicial y estado final.
* **Funciones**: Establecen una relación entre los elementos de los conjuntos A (dominio) y B (codominio).



* En cuanto a su **utilización:**
* **Acciones**: Describen acciones complejas en función de otras más elementales o primitivas. Se utilizan de idéntico modo a las primitivas.
* **Funciones**: Extienden el repertorio de operadores definiendo otros nuevos. Se utilizan del mismo modo que los operadores, e.d., en expresiones donde pueda aparecer un valor del tipo correspondiente (codominio).
* En cuanto a su **descripción:**
* **Acciones**: Su algoritmo establece la forma en que acciones más elementales se combinan para modificar el estado del proceso y conseguir su postcondición a partir de su precondición.
* **Funciones**: También se describen en forma de algoritmo, pero el objetivo de éste no es modificar el estado del proceso sino calcular el elemento del tipo resultado (codominio) que se corresponde con los valores de los parámetros suministrados (dominio).
* **Descripción:**
* Como puede verse en los ejemplos, una función se describe mediante su propio algoritmo.
* Desde este punto de vista, una función puede visualizarse como una acción en la que:
* Todos los parámetros son DATO
* Produce un único resultado que es asignado al nombre de la función como último paso de ésta. El identificador del nombre de la función se considera dentro de ésta como una variable de sólo escritura.

1. *Estructuración de información: tablas*

****

****