UNIDAD 4: SUBPROGRAMAS. FUNCIONES Y PROCEDIMIENTOS

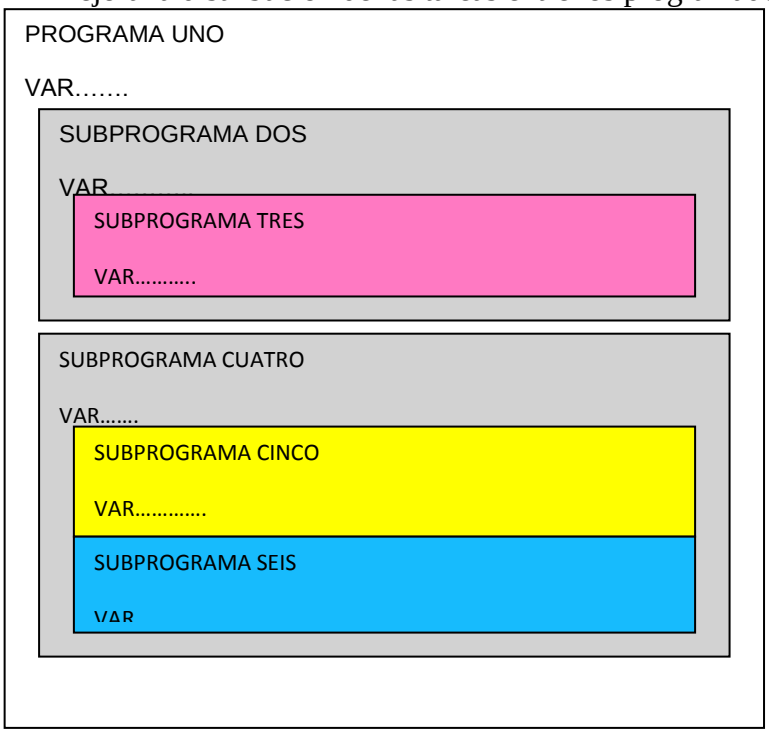
SUBPROGRAMAS

Lo que hemos visto hasta ahora es la resolución de problemas dentro de una única estructura de control. Pero muchas veces tenemos que resolver problemas que para manejarlo mejor lo debemos **dividir en subproblemas**. Ahí es donde usamos subprogramas **que nos permiten organizar los problemas complejos.**

Otra situación es que una secuencia de acciones debe repetirse varias veces, en distintos lugares de un programa. Entonces para no escribir lo mismo otra vez podemos **agrupar las acciones en bloques denominados subprogramas.**

BENEFICIOS DE SUBPROGRAMAS

- Menor escritura
- Mayor claridad en la lectura
- Mayor facilidad para modificar sin alterar todo el programa
- Mayor posibilidad de reutilización, incluso con otros programadores.
- Mejora la distribución de las tareas entre los programadores.



El programa principal UNO contiene a los subprogramas DOS y CUATRO. A su vez el subprograma DOS contiene al subprograma TRES. Y el CUATRO contiene a CINCO y SEIS.

El programa que contiene subprogramas ejecuta sus instrucciones y en algún momento hace una llamada al subprograma por su nombre, y le transfiere el control. Cuando se terminan de ejecutar todas las instrucciones del subprograma, regresa el control al punto donde se suspendió la ejecución de las instrucciones del programa y continúa normalmente.

ENCABEZAMIENTO DE SUBPROGRAMAS

Encabezamiento que contiene:

- Tipo de subprograma: Procedimiento o Función.
- Nombre de subprograma
- Lista de parámetros o argumentos que posibilitan la comunicación entre el subprograma y el programa que lo invoca.

- A través de la lista de parámetros se pasan los valores con los que debe trabajar el subprograma. Estos valores pueden ser constantes, variables, expresiones o funciones.
- La correspondencia entre los valores de la llamada, y los del subprograma, son uno a uno en la misma posición, del mismo tipo de dato y la misma cantidad.
- A los parámetros que se usan en el programa que llama se denominan **parámetros reales o argumentos de llamada.**
- A los parámetros que se usan en el subprograma se denominan **parámetros formales.**

TIPOS DE SUBPROGRAMAS

- Podemos encontrar procedimientos y funciones.
- Tanto los procedimientos como las funciones son subprogramas que **ejecutan una determinada tarea**. En base a las características de la tarea a resolver se decide hacer un procedimiento o una función.

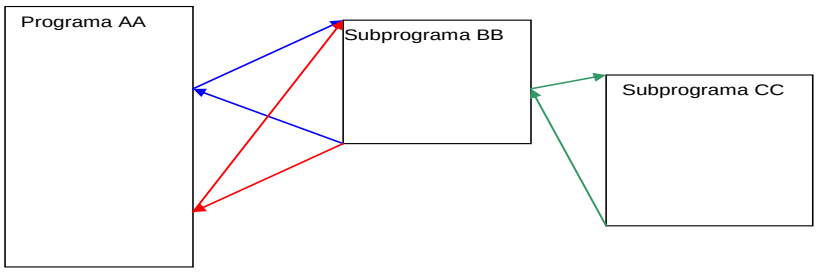
COMPARACIÓN

	PROCEDIMIENTO	**FUNCIÓN**
Cantidad de Resultados	0 .. n	1
Lugar donde se devuelve el resultado	Parámetros	Nombre de la función
Forma de llamarla	Línea de programa	Expresión
Asignación de lectura y escritura	Sí permite	No permite

- ✓ Si el subproblema tiene que devolver **un solo resultado** y **no debe realizar operaciones de escritura/lectura** es conveniente hacer **una función.**
- ✓ Si se necesita **operaciones de escritura/lectura** o bien **ninguno o más de un resultado** hacer **un procedimiento.**

LOS PARÁMETROS PUEDEN SER:

- **Entrada:** los que el programa envía al subprograma**.**
- **Salida:** los que el subprograma devuelve con un determinado valor.
- **Entrada/Salida**: los que el subprograma modifica.



PROCEDIMIENTOS

Cómo se escribe el procedimiento:

PROCEDIMIENTO nombre (lista parámetros formales)
VAR
(declaración de variables locales)
INICIO
    Acciones

FINPROCEDIMIENTO

Donde:

- Nombre: nombre para referirse al procedimiento, se respetan las mismas pautas que para nombres de programas.
- Lista parámetros formales: identifica nombre y tipo de datos de las variables que usamos para recibir los valores con los que debe trabajar o en los cuales debe devolverse los resultados.
- VAR: declaración de **variables locales**, es decir sólo tienen validez en el procedimiento.
- Acciones: conjunto de **instrucciones** que forman el cuerpo del procedimiento.
- FINPROCEDIMIENTO: momento en que se devuelve el control al programa o subprograma que lo llamó.
- Para llamarlo desde el programa principal o desde otro subprograma**, en una sentencia de programa:**
  nombre (lista parámetros reales)

FUNCIONES

Cómo se escribe la función:

FUNCIÓN nombre (lista parámetros formales): tipo de resultado
VAR
(declaración de variables locales)
INICIO
Acciones
nombre = expresión/variable
RETORNO

Donde:

- Nombre: nombre para referirse a la función, se respetan las mismas pautas que para nombres de programas.
- Lista parámetros formales: identifica nombre y tipo de datos de las variables que usamos para recibir los valores con los que debe trabajar.
- Tipo de resultado: indica el tipo de dato del valor que devuelve la función al punto que la llamaron (en el nombre).
- VAR: declaración de **variables locales**, es decir sólo tienen validez en la función.
- Acciones: conjunto de **instrucciones** que forman el cuerpo de la función.
- RETORNO: momento en que se devuelve el control al programa o subprograma que la llamó.
- Para llamar desde el programa principal o desde otro subprograma:
  nombre (lista parámetros reales)
  Desde:
  - Una expresión
  - La parte derecha de una asignación
  - En una lista de argumentos de llamada

PARÁMETROS

- A través de la lista de parámetros **se pasan los valores o las direcciones** donde se encuentran los valores, con los que debe trabajar el subprograma.
- Es importante determinar qué método se utilizará para pasar los parámetros, ya que de esto depende los resultados de los programas.

CLASIFICACIÓN DE PARÁMETROS

- **Entrada:** proporcionan los valores desde el programa o subprograma que llama.

- **Salida:** valores resultantes del subprograma. Las funciones no usan parámetros de salida, ya que el resultado lo devuelven en el nombre.
- **Entrada/Salida:** un mismo parámetro sirve para mandar argumentos a un subprograma y para devolver el resultado.

RECORDAR

- La correspondencia entre los valores de la llamada y los del subprograma.
- A los parámetros del programa que llama se denominan parámetros reales.
- A los parámetros del subprograma se denominan parámetros formales.

PASAJE DE PERÁMETROS

Los métodos más empleados para realizar el pasaje de parámetros son:

- **Por Valor o Protegido**
- **Por referencia**

POR VALOR O PROTEGIDO

- Es la forma más simple. Se comunican los programas enviando una **copia de la variable original.**
- El subprograma recibe los valores de los parámetros, como valores iniciales, ejecuta sus acciones, pero no los devuelve con otros valores. Si sufre alguna modificación no afecta la variable original del programa principal.
- **Las funciones siempre utilizan parámetros por valor y devuelve el resultado en el nombre de la función.**

EJEMPLO DE PASAJE POR VALOR

```
PROGRAMA pares
VAR nro: ENTERO
INICIO
Escribir ("Ingrese un número")
Leer (nro)
SI nro > 0 entonces
        SI par (nro) ENTONCES
          Escribir ("PAR")
        SINO
          Escribir ("IMPAR")
        FINSI
SINO
  Escribir(" NEGATIVO o CERO")
FINSI
FINPROGRAMA

FUNCIÓN par (num:ENTERO): LÓGICO
VAR cociente, resto: ENTERO
INICIO
// verificar si un número es par
cociente  = num / 2
resto = num – cociente * 2
SI resto = 0 ENTONCES
        par  = [V]
SINO
        par  = [F]
FINSI
RETORNO
```

POR REFERENCIA

- Se comunican los programas, pero en lugar de enviar una copia de la variable original, en el parámetro se escribe la dirección de la misma.
- El subprograma **recibe la dirección de las variables**, por lo tanto todo lo que ejecute o modifique sobre el parámetro formal recibido, se están realizando directamente en la variable original, ya que es adonde apunta el parámetro formal.
- Se debe **anteponer** en la lista de parámetros formales **"por ref."** al nombre de la variable para indicar que es por referencia.

EJEMPLO DE PASAJE POR REFERENCIA

```
PROGRAMA valorPositivo
VAR nro: ENTERO
INICIO
// llamar procedimiento
positivo (nro)
Escribir (" Número =", nro)
FINPROGRAMA

PROCEDIMIENTO positivo (por ref. num: ENTERO)
VAR
INICIO
// ingresa el número
ITERAR
   Escribir ("Ingrese un número positivo")
   Leer (num)
SALIRSI num > 0
    Escribir ("El número no es positivo")
FINITERAR
FINPROCEDIMIENTO
```

VARIABLES GLOBALES Y LOCALES

- Las variables que se utilizan en programas principales y subprogramas se clasifican según el ámbito donde se definen en:
  - **Globales**
  - **Locales**

- ***Globales:*** se declaran y definen en el programa principal y **su alcance es amplio**, pues tiene influencia tanto en el programa principal como todos los subprogramas.
- ***Locales:*** se declaran y definen en el subprograma, pero **existe sólo dentro del subprograma** y desaparece cuando se regresa el control al programa principal.
- Variables

Definidas en:	Accesibles desde:
UNO	TODOS
DOS	DOS Y TRES
TRES	TRES
CUATRO	CUATRO, CINCO, SEIS

CINCO CINCO

SEIS SEIS