Funciones/Subprocesos en PSeInt

Si el perfil de lenguaje seleccionado lo permite (ver Opciones del Pseudocódigo), se pueden declarar nuevas funciones o subprocesos en un algoritmo en Pseudocódigo. La sintaxis para ello es la siguiente:

```
SubProceso Tipo  variable_de_retorno <- nombre_de_la_funcion ( Tipo argumento_1, Tipo argumento_2, ... )
    acción 1;
    acción 1;
        .
        .
        .
    acción n;
FinSubproceso
```

Comienza con la palabra clave **SubProceso** (o Función, son equivalentes) seguida del Tipo de la variable de retorno (real, entero, string, etc.), el nombre de la variable de retorno, el signo de asignación, el nombre del subproceso, y finalmente, la lista de argumentos entre paréntesis con su correspondiente Tipo antes del nombre. Existen variantes para esta estructura. Si la función no retorna ningún valor, pueden omitirse el identificador variable_de_retorno y el signo de asignación, es decir, colocar directamente el nombre y los argumentos a continuación de la palabra clave SubProceso. Si el subproceso no recibe ningún valor pueden colocarse los paréntesis vacíos u omitirse, finalizando la primer línea con el nombre del subproceso. Las reglas para los nombres de subprocesos, variables de retorno y argumentos son las mismas que para cualquier identificador en pseudocódigo.

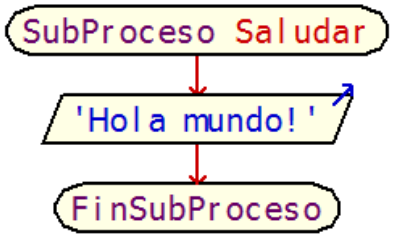
Además, opcionalmente pueden agregarse las palabras claves Por Valor o Por Referencia para indicar el tipo de pasaje en cada argumento. Si no se indica, los arreglos se pasan por referencia, las demás expresiones por valor. El pasaje por referencia implica que si la función modifica el argumento, se modificará en realidad la variable que se utilizó en la llamada, mientras que el pasaje por valor implica que la función opera con una copia de la variable (o el resultado de la expresión) que se utilizó en la llamada, por lo que las modificaciones que aplique la función no se verán reflejadas fuera de la misma.

Para invocar a la función se debe utilizar su nombre y entre paréntesis los parámetros, que podrán ser expresiones sólo si el tipo de pasaje es por referencia. Una llamada puede ser en sí una instrucción, pero si la función retorna algún valor, también puede utilizarse como operando dentro de una expresión.

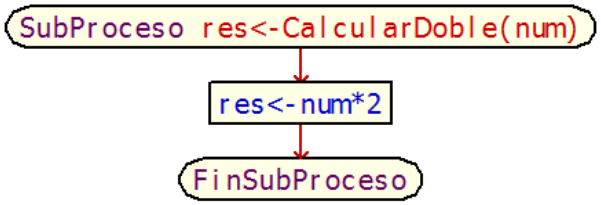
El ejemplo Subprocesos, muestra un programa que declara dos funciones, una que retorna un valor y se que es luego utilizado dentro de una expresión para mostrar en pantalla, y otra que no recibe argumentos; el ejemplo Potencia, muestra una función recursiva; y el ejemplo Promedio muestra una función que recibe un arreglo como argumento.

Ejemplos de varias funciones y procedimientos que se llaman desde un programa principal.

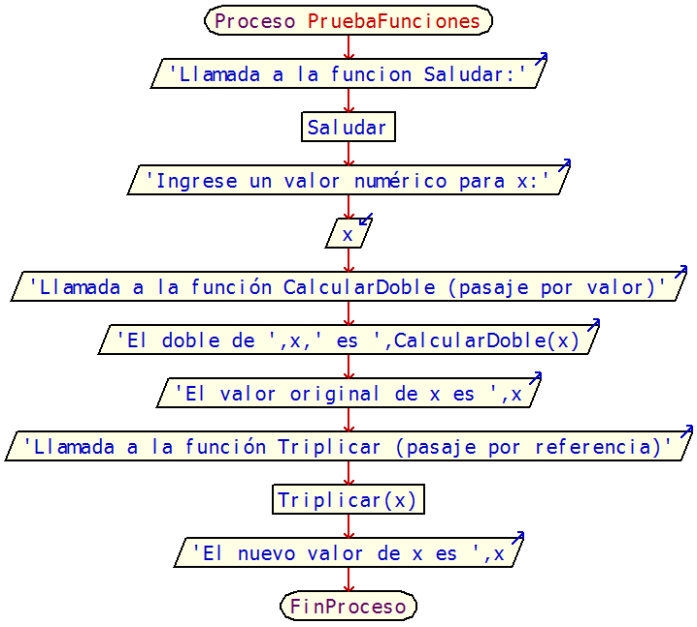
```
// funcion que no recibe ni devuelve nada
SubProceso Saludar
	Escribir "Hola mundo!"
FinSubProceso
```




```
// funcion que recibe un argumento por valor, y devuelve su doble
SubProceso Real res <- CalcularDoble(Real num)
	res <- num*2 // retorna el doble
FinSubProceso
```




```
// funcion que recibe un argumento por referencia, y lo modifica
SubProceso Triplicar(real num por referencia)
	num <- num*3 // modifica la variable duplicando su valor
FinSubProceso


// proceso principal, que invoca a las funciones antes declaradas
Proceso PruebaFunciones
	Definir x como real

	Escribir "Llamada a la funcion Saludar:"
	Saludar // como no recibe argumentos pueden omitirse los paréntesis vacios

	Escribir "Ingrese un valor numérico para x:"
	Leer x

	Escribir "Llamada a la función CalcularDoble (pasaje por valor)"
	Escribir "El doble de ",x," es ", CalcularDoble(x)
	Escribir "El valor original de x es ",x

	Escribir "Llamada a la función Triplicar (pasaje por referencia)"
	Triplicar(x)
	Escribir "El nuevo valor de x es ", x

FinProceso
```


Proceso PruebaFunciones
'Llamada a la funcion Saludar:'
Saludar
'Ingrese un valor numérico para x:'
x
'Llamada a la función CalcularDoble (pasaje por valor)'
'El doble de ',x,' es ',CalcularDoble(x)
'El valor original de x es ',x
'Llamada a la función Triplicar (pasaje por referencia)'
Triplicar(x)
'El nuevo valor de x es ',x
FinProceso