NOMBRE DE LA MATERIA: **PROGRAMACIÓN**

NRC: **42555**

HORARIO: **MARTES Y JUEVES 9 AM – 10:55AM**

ESTUDIANTE: **EFRAIN ROBLES PULIDO**

CODIGO: **221350095**

TEMA: **FUNCIONES**

FECHA: **7 DE NOVIEMBRE DE 2021**

Funciones

La programación modular es una técnica que consiste en separar un problema en las diferentes tareas que se quieren resolver, dando origen a la creación de módulos (pequeños programas a los que llamaremos funciones), donde cada módulo o función se diseña, se codifica y se procesa de manera independiente.

El uso de las funciones hace la programación más fácil y eficiente pues permite:

- Reducir la complejidad del programa (“divide y venceras”).
- Eliminar código duplicado.
- Controlar fácilmente los efectos de los cambios.
- Ocultar detalles de implementación.
- Reutilizar código.
- Facilitar la legibilidad del código.

Las funciones en C no se pueden anidar. Esto significa que una función no se puede declarar dentro de otra función. La razón para esto es permitir un acceso muy eficiente a los datos. Entonces, todas las funciones son externas o globales, es decir, pueden ser llamadas desde cualquier punto del programa.

Una función consta de una **cabecera** que comienza con el tipo del valor devuelto por la función, seguido del nombre y argumentos de dicha función. A continuación, va el cuerpo de la función, que es un conjunto de sentencias cuya ejecución hará que se resuelva el problema para el que está diseñada la función. Esto determina el valor particular del resultado que ha de devolverse al programa llamador.

Los aspectos más sobresalientes en el diseño de una función son:

- **Tipo de resultado.** Es el tipo de dato que devuelve la función C y aparece antes del nombre de la función.
- **Lista de parámetros**. Es una lista de parámetros tipificados (con tipos) que utilizan el formato siguiente: tipo1 parámetro1, tipo2 parámetro2, ...
- **Cuerpo de la función**. Se encierra entre llaves de apertura ({) y cierre (}). No hay punto y coma después de la llave de cierre.
- **Paso de parámetros**. Posteriormente se verá que el paso de parámetros en C se hace siempre por valor.
- No se pueden declarar funciones anidadas.
- **Declaración local**. Las constantes, tipos de datos y variables declaradas dentro de la función son locales a la misma y no perduran fuera de ella.
- **Valor devuelto por la función**. Mediante la palabra reservada return se devuelve el valor calculado por la función.

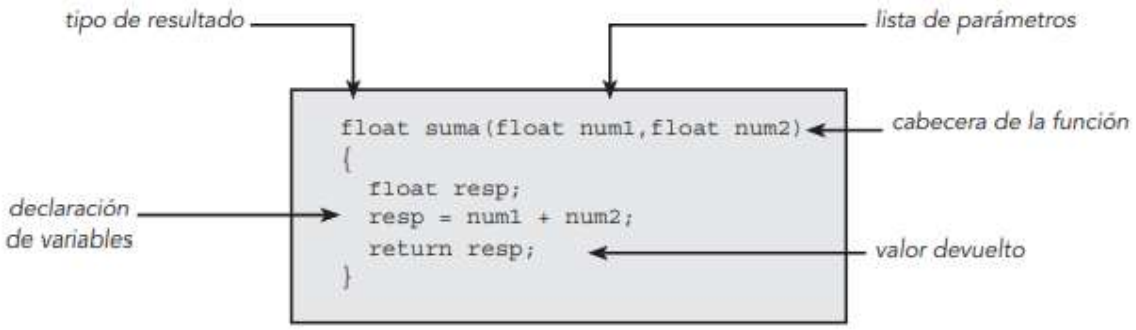
```
tipo_de_retorno nombreFunción (listaDeParámetros)
{
  cuerpo de la función
  return expresión
}
```

tipo_de_retorno ← Tipo de valor devuelto por la función o la palabra reservada **void** si la función no devuelve ningún valor.

nombreFunción ← Identificador o nombre de la función.

listaDeParámetros ← Lista de declaraciones de los parámetros de la función separados por comas.

expresión ← Valor que devuelve la función.



Una llamada a la función produce la ejecución de las sentencias del cuerpo de la función y un retorno a la unidad de programa llamadora después de que la ejecución de la función se haya terminado, normalmente cuando se encuentra una sentencia return.

Nombre de una función

Comienza con una letra o un subrayado (_) y puede contener tantas letras, números o subrayados como desee.

Tipo de dato de retorno

Si la función no devuelve un valor int, se debe especificar el tipo de dato devuelto la función; cuando devuelve un valor int, se puede omitir ya que en forma predeterminada C supone que todas las funciones son enteras. A pesar de ello siempre conviene especificar el tipo, aun siendo de tipo int, para mejor legibilidad. El tipo debe ser uno de los tipos simples de C, como int, char o float, o un apuntador (puntero) a cualquier tipo C, o un tipo struct.

Si una función no devuelve un resultado, se utiliza el tipo void, que se considera como un tipo de dato especial. La razón es que se utilizan como subrutinas para realizar una tarea concreta. Una función que no devuelve un resultado, a veces se denomina procedimiento. Para indicar al compilador que una función no devuelve resultado, se utiliza el tipo de retorno void.

Resultados de una función

Una función devuelve un único valor. El valor devuelto (expresión) puede ser cualquier tipo de dato conocido por el lenguaje C (tipo simple, o tipo estructurado). Se pueden retornar valores múltiples devolviendo un apuntador a una estructura o un array (arreglo). El valor de retorno debe seguir las mismas reglas que se aplican a un operador de asignación. Sin embargo, si se devuelve un int y el tipo de retorno es un float, se realiza la conversión automáticamente.

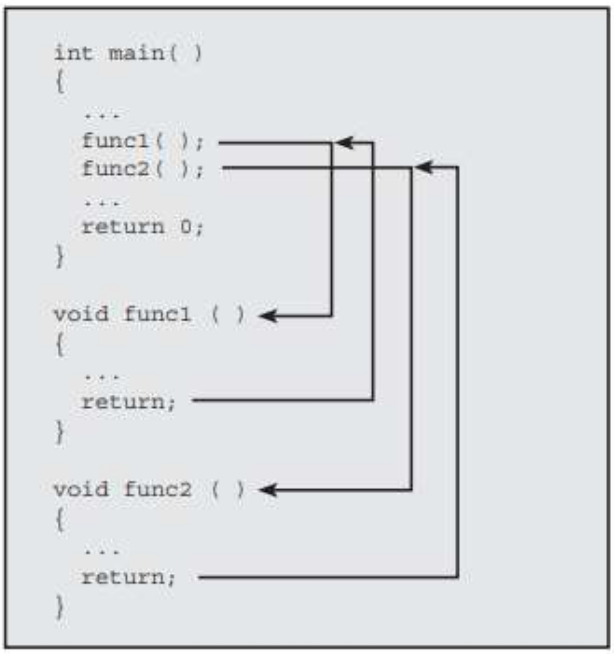
El valor devuelto se suele encerrar entre paréntesis, pero su uso es opcional. En algunos sistemas operativos, la función principal (main()) puede devolver un resultado al entorno llamador.

Llamada a una función

Las funciones, para poder ser ejecutadas, han de ser llamadas o invocadas. Cualquier expresión puede contener una llamada a una función que redirigirá el control del programa a la función nombrada. Normalmente la llamada a una función se realizará desde la función principal main(), aunque naturalmente también podrá ser desde otra función.

La función llamada que recibe el control del programa se ejecuta desde el principio y cuando termina (se alcanza la sentencia return o la llave de cierre [}] si se omite return) el control del programa vuelve y retorna a la función main() o a la función llamadora si no es main().

Se puede llamar a una función y no utilizar el valor que se devuelve. En esta llamada a función: func() el valor de retorno no se considera. El formato func() sin argumentos es el más simple. Para indicar que la llamada a una función no tiene argumentos se sitúa una palabra reservada void entre paréntesis en la declaración de la función y posteriormente en lo que se denominará **prototipo**.



Bibliografía:

Márquez G., Osorio S., Olvera N. (2011). Introducción a la Programación Estructurada en C. Pearson

Joyanes Aguilar, L., (2014). Programación en C, C++, Java y UML. McGraw Hill. 2a. Edición

Programación en C ++ / Funciones - Wikilibros. (2021). Consultado el 6 de noviembre de 2021 en https://es.wikibooks.org/wiki/Programaci%C3%B3n_en_C%2B%2B/Funciones