Lenguaje de Programación: C++ Funciones

José Luis Alonzo Velázquez

Universidad de Guanajuato

Octubre 2010

Declaración de funciones

Antes de escribir una función es necesario informarle al Compilador los tamaños de los valores que se le enviarán en el stack y el tamaño de los valores que ella retornará al programa invocante . Estas informaciones están contenidas en la **declaración** del **prototipo** de la **función**. Formalmente dicha declaración queda dada por :

```
tipo del valor de retorno   nombre_de_la_función(lista de tipos de parámetros)
```

Declaración de funciones

Antes de escribir una función es necesario informarle al Compilador los tamaños de los valores que se le enviarán en el stack y el tamaño de los valores que ella retornará al programa invocante . Estas informaciones están contenidas en la **declaración** del **prototipo** de la **función**. Formalmente dicha declaración queda dada por :

```
tipo del valor de retorno   nombre_de_la_función(lista de tipos de parámetros)
```

Ejemplo

```
float mi_funcion(int i, double j ) ;

double otra_funcion(void) ;

otra_mas(long p) ;

void  la_ultima(long double z,char y,int x,unsigned long w);
```

Observaciones

El primer término del prototipo da, como hemos visto el tipo del dato retornado por la función; en caso de obviarse el mismo se toma, por omisión, el **tipo int**. Sin embargo, aunque la función devuelva este tipo de dato, para evitar malas interpretaciones es conveniente explicitarlo .

Observaciones

El primer término del prototipo da, como hemos visto el tipo del dato retornado por la función; en caso de obviarse el mismo se toma, por omisión, el **tipo int**. Sin embargo, aunque la función devuelva este tipo de dato, para evitar malas interpretaciones es conveniente explicitarlo .

Ya que el "default" del tipo de retorno es el **int**, debemos indicar cuando la función NO retorna nada, esto se realiza por medio de la palabra **VOID** (sin valor). De la misma manera se actúa, cuando no se le enviarán argumentos. Más adelante se profundizará sobre el tema de los argumentos y sus características.

Observaciones

La declaración debe anteceder en el programa a la definición de la función. Es normal, por razones de legibilidad de la documentación, encontrar todas las declaraciones de las funciones usadas en el programa, en el **HEADER** del mismo, junto con los include de los archivos *.h que tienen los prototipos de las funciones de Librería.

Observaciones

La declaración debe anteceder en el programa a la definición de la función. Es normal, por razones de legibilidad de la documentación, encontrar todas las declaraciones de las funciones usadas en el programa, en el **HEADER** del mismo, junto con los include de los archivos *.h que tienen los prototipos de las funciones de Librería.

Si una ó más de nuestras funciones es usada habitualmente, podemos disponer su prototipo en un archivo de texto, e incluirlo las veces que necesitemos, según se vera más adelante.

Definición de las funciones

La definición de una función puede ubicarse en cualquier lugar del programa, con sólo dos restricciones: debe hallarse luego de dar su prototipo, y no puede estar dentro de la definición de otra función (incluida main()). Es decir que a diferencia de Pascal, en C las definiciones no pueden anidarse.

Definición de las funciones

La definición de una función puede ubicarse en cualquier lugar del programa, con sólo dos restricciones: debe hallarse luego de dar su prototipo, y no puede estar dentro de la definición de otra función (incluida main()). Es decir que a diferencia de Pascal, en C las definiciones no pueden anidarse.

NOTA

No confundir definición con llamada; una función puede llamar a tantas otras como desee. La definición debe comenzar con un encabezamiento, que debe coincidir totalmente con el prototipo declarado para la misma, y a continuación del mismo, encerradas por llaves se escribirán las sentencias que la compone.

```
#include <stdio.h>
/* Declaración observe que termina en  ";"  */
float mi_funcion(int i, double j );
int main(){
float k ;
int p ;
double z ;
...........
k = mi_funcion( p, z ); /* LLAMADA a la función */
...........
}
/* fin de la función main() */
/* Definición  observe que NO lleva  ";"  */
float mi_funcion(int i, double j ){
float n;
..................
printf("%d", i ); /* LLAMADA  a otra función */
..................
return ( 2 * n ); /* RETORNO devolviendo un valor float */
}
```

Problema para clase

Hacer una función que manda llamar otra función que a su ves manda llamar otra función.

Problema extra clase

Utilizar un archivo header y ver como funciona para poder usarlo en varios proyectos.

 Programming Principles and Practice Using C++, Bjarne Stroustrup.

 http://www.codeblocks.org

 http://www.wxwidgets.org

 (O'Reilly) Practical C Programming (3rd Edition)