

# Lenguaje de Programación: C++ Funciones

José Luis Alonzo Velázquez

Universidad de Guanajuato

Septiembre 2010

## ¿Que es una función?

Una función es un conjunto de líneas de código que realizan una tarea específica y puede retornar un valor. Las funciones pueden tomar parámetros que modifiquen su funcionamiento. Las funciones son utilizadas para descomponer grandes problemas en tareas simples y para implementar operaciones que son comúnmente utilizadas durante un programa y de esta manera reducir la cantidad de código. Cuando una función es invocada se le pasa el control a la misma, una vez que esta finalizó con su tarea el control es devuelto al punto desde el cual la función fue llamada.

## Sintaxis

```
<tipo> [clase::] <nombre> ( [Parámetros] )  
{  
    cuerpo;  
}
```

## Sintaxis

```
<tipo> [clase::] <nombre> ( [Parámetros] )  
{  
    cuerpo;  
}
```

## Ejemplo

```
// regresar el cuadrado de un número  
double cuadrado(double n)  
{  
    return n*n;  
}
```

## Parámetros por valor

La función `cuadrado()` es un clásico ejemplo que muestra el paso de parámetros por valor, en ese sentido la función `cuadrado()` recibe una copia del parámetro  $n$ . En la misma función se puede observar que se realiza un cálculo ( $n * n$ ), sin embargo el parámetro original no sufrirá cambio alguno, esto seguirá siendo cierto aún cuando dentro de la función hubiera una instrucción parecida a  $n = n * n$ ; o  $n* = n$ ;

## Parámetros por valor

La función `cuadrado()` es un clásico ejemplo que muestra el paso de parámetros por valor, en ese sentido la función `cuadrado()` recibe una copia del parámetro  $n$ . En la misma función se puede observar que se realiza un cálculo ( $n * n$ ), sin embargo el parámetro original no sufrirá cambio alguno, esto seguirá siendo cierto aún cuando dentro de la función hubiera una instrucción parecida a  $n = n * n$ ; o  $n* = n$ ;

## Ejemplo

```
// regresar el cuadrado de un número
double cuadrado(double n)
{
    return n*n;
}
```

## Ejemplo

```
// regresar el cuadrado de un número
double cuadrado2(double &n)
{
    n *= n;
    return n;
}
```

## Ejemplo

```
// regresar el cuadrado de un número
double cuadrado2(double &n)
{
    n *= n;
    return n;
}
```

## Parámetros por referencia

La función `cuadrado2()` es un clásico ejemplo que muestra el paso de parámetros por referencia, en ese sentido la función `cuadrado2()` recibe el parámetro  $n$ . En la misma función se puede observar que se realiza un calculo ( $n * n$ ), sin embargo el parámetro original sufrirá cambio, esto seguirá siendo cierto aún cuando dentro de la función hubiera una instrucción parecida a  $n = n * n$ ; o  $n* = n$ ;



## Parámetros constantes

Los parámetros usados por una función pueden declararse como constantes ( `const` ) al momento de la declaración de la función. Un parámetro que ha sido declarado como constante significa que la función no podrá cambiar el valor del mismo ( sin importar si dicho parámetro se recibe por valor o por referencia ).

## Parámetros constantes

Los parámetros usados por una función pueden declararse como constantes ( `const` ) al momento de la declaración de la función. Un parámetro que ha sido declarado como constante significa que la función no podrá cambiar el valor del mismo ( sin importar si dicho parámetro se recibe por valor o por referencia ).

## Ejemplo

```
int funcionX( const int n );  
void printstr( const char *str );
```

## Parámetros con valor por defecto

Los parámetros usados por una función pueden declararse con un valor por defecto. Un parámetro que ha sido declarado con valor por defecto es opcional a la hora de hacer la llamada a la función.

## Parámetros con valor por defecto

Los parámetros usados por una función pueden declararse con un valor por defecto. Un parámetro que ha sido declarado con valor por defecto es opcional a la hora de hacer la llamada a la función.

## Ejemplo

```
void saludo( char* mensaje = "Hola sudafrica 2010" );
```

la misma puede ser invocada como:

```
saludo(); // sin parámetro
```

```
saludo("Sea usted bienvenido a C++"); // con parámetro
```

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

 <http://www.codeblocks.org>

 <http://www.wxwidgets.org>

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