



**República Bolivariana de Venezuela**  
**Aldea Universitaria Liceo Fray Pedro de Agreda**  
**Trayecto I**  
**Introducción a la Programación III**

**Fac. Elías Cisneros**  
**Fac. Juan Cisneros**

**Lenguaje de programación C y C++**  
*Material extraído de Wikipedia, la enciclopedia libre*

**Contenido**

|  |   |
|--|---|
| Antecedentes.....  | 1 |
| Filosofía.....   | 2 |
| ANSI C e ISO C.....  | 2 |
| Breve Reseña de C++.....   | 3 |
| Algunos lenguajes de programación y su clasificación.....          | 3 |
| Estructura básica de un programa en C++.....                       | 4 |
| Compilación de un programa escrito en C++ en plataforma Linux..... | 6 |

**Antecedentes**

C es un lenguaje de programación creado en 1969 por Ken Thompson y Dennis M. Ritchie en los Laboratorios Bell como evolución del anterior lenguaje B. Al igual que B, es un lenguaje orientado a la implementación de Sistemas Operativos, concretamente Unix. C es apreciado por la eficiencia del código que produce y es el lenguaje de programación más popular para crear software de sistemas, aunque también se utiliza para crear aplicaciones.

Se trata de un lenguaje débilmente tipado de medio nivel pero con muchas características de bajo nivel. Es necesario aclarar que un lenguaje fuertemente tipado: es un lenguaje en el que los tipos de datos se mantienen siempre. Java y Python son fuertemente tipados. Si se tiene un entero, no se le puede tratar como una cadena sin convertirlo explícitamente .

Lenguaje débilmente tipado es un lenguaje en el que los tipos pueden ignorarse; lo contrario de fuertemente tipado. VBScript es débilmente tipado. En VBScript, se puede concatenar la cadena '12' con el entero 3 para obtener la cadena '123', y después tratarla como el entero 123, todo ello sin conversión explícita.



## Filosofía

Es muy posible escribir C a bajo nivel de abstracción; de hecho, C se usó como intermediario entre diferentes lenguajes. En parte a causa de ser de relativamente bajo nivel y de tener un conjunto de características modesto, se pueden desarrollar compiladores de C fácilmente.

C tiene las siguientes características de importancia:

- Un núcleo del lenguaje simple, con funcionalidades añadidas importantes, como funciones matemáticas y de manejo de ficheros, proporcionadas por bibliotecas.
- Es un lenguaje muy flexible que permite programar con múltiples estilos. Uno de los más empleados es el estructurado.
- Un sistema de tipos que impide operaciones sin sentido.
- Usa un lenguaje de preprocesado, el preprocesador de C, para tareas como definir macros e incluir múltiples ficheros de código fuente.
- Acceso a memoria de bajo nivel mediante el uso de punteros.
- Un conjunto reducido de palabras clave.
- Los parámetros se pasan por valor. El paso por referencia se puede simular pasando explícitamente el valor de los punteros.
- Punteros a funciones y variables estáticas, que permiten una forma rudimentaria de encapsulado y polimorfismo.
- Tipos de datos agregados (`struct`) que permiten que datos relacionados se combinen y se manipulen como un todo.

C es más eficiente que otros lenguajes. Típicamente, sólo la programación cuidadosa en lenguaje ensamblador produce un código más rápido, pues da control total sobre la máquina, aunque los avances en los compiladores de C y la complejidad creciente de los procesadores modernos han reducido gradualmente esta diferencia. En 1973, el lenguaje C se había vuelto tan potente que la mayor parte del kernel Unix, originalmente escrito en el lenguaje ensamblador PDP-11/20, fue reescrita en C. Éste fue uno de los primeros núcleos de sistema operativo implementados en un lenguaje distinto al ensamblador.

## ANSI C e ISO C

A finales de la década de 1970, C empezó a sustituir a BASIC como lenguaje de programación de microcomputadores predominante. Durante la década de 1980 se empezó a usar en los IBM PC, lo que incrementó su popularidad significativamente. Al mismo tiempo, Bjarne Stroustrup empezó a trabajar con algunos compañeros de Bell Labs para añadir funcionalidades de programación orientada a objetos a C. El lenguaje que crearon, llamado C++, es hoy en día el lenguaje de programación de aplicaciones



más común en el sistema operativo Microsoft Windows; mientras que C sigue siendo más popular en el entorno Unix.

También se han creado numerosos lenguajes inspirados en la sintaxis de C, pero que no son compatibles con él:

- Java, que une la sintaxis del C++ a una orientación a objetos más similar a la de Smalltalk y Objective C.
- JavaScript, un lenguaje de scripting creado en Netscape e inspirado en la sintaxis de Java diseñado para dar a las páginas web mayor interactividad. A la versión estandarizada se la conoce como ECMAScript.
- C# (pronunciado *C Sharp*) es un lenguaje desarrollado por Microsoft derivado de C/C++ y Java.

### Breve Reseña de C++

En los laboratorios de AT&T Bell, que Bjarne Stroustrup diseñó y desarrolló C++ buscando un lenguaje con las opciones de programación orientada a objetos. En ese entonces el desarrollo del estándar de C++ acaparaba la atención de los diseñadores. En el año 1995, se incluyeron algunas bibliotecas de funciones al lenguaje C. Y con base en ellas, se pudo en 1998 definir el estándar de C++.

Es un mito pensar que entonces C++ desplazó a C, algunas soluciones a problemas requieren de la estructura simple de C más que la de C++, C generalmente es usado por comodidad para escribir controladores de dispositivos y para programas de computadoras con recursos limitados.

C++ proporciona orientación a objetos, esta versión combina la flexibilidad y el acceso de bajo nivel de C con las características de la programación orientada a objetos como abstracción, encapsulación y ocultación. Una consideración importante es que hasta la publicación de este estándar, C había sido mayormente un subconjunto estricto del C++. Era muy sencillo "actualizar" un programa de C hacia C++ y mantener ese código compilable en ambos lenguajes. Sin embargo, el nuevo estándar agrega algunas características que C++ no admite.

### Algunos lenguajes de programación y su clasificación

| Alto nivel                         | Nivel medio               | Nivel bajo  |
|------------------------------------|---------------------------|-------------|
| Ada<br>Modula-2<br>Pascal<br>Cobol | C++<br>Java<br>C<br>FORTH | Ensamblador |



|                  |                  |  |
|------------------|------------------|--|
| FORTRAN<br>Basic | Macroensamblador |  |
|------------------|------------------|--|

## Estructura básica de un programa en C++

|   |                                       |
|---|---------------------------------------|
| <b>using namespace std;</b><br><b>#include &lt;iostream&gt;</b> | Declaración de librerías              |
| <b>int main(void)</b>   | Función main o principal              |
| {   | Llaves de apertura de la función main |
| <b>cout&lt;&lt;"Me gusta la programación"&lt;&lt;endl;</b>      | Secuencia de instrucciones            |
| <b>return 0;</b>  | Valor de retorno de la función        |
| }   | Llaves de cierre de la función        |

## Análisis del código fuente

```
using namespace std;  
#include <iostream>
```

La parte del `#include` se refiere a la biblioteca de funciones que vamos a importar o utilizar. Es decir para llamar a una biblioteca en particular debemos hacer lo siguiente:

```
#include <librería_solicitada>
```

El estándar de C++ incluye varias bibliotecas de funciones, y dependiendo del compilador que se esté usando.

```
int main(void){
```

Todo programa en C++ comienza con una función `main()`, y sólo puede haber una. En C++ el `main()` siempre regresa un entero, es por eso se antepone "int" a la palabra "main". Los paréntesis que le siguen contienen lo que se le va a pasar a la función. En



este caso se puso la palabra “void” que significa vacío, es decir que a la función main no se le está mandando ningún parámetro, podría omitirse el void dentro de los paréntesis, el compilador asume que no se enviará nada. La llave que se abre significa que se iniciará un bloque de instrucciones.

```
cout<<”hola mundo”<<endl;
```

Esta es una instrucción. La instrucción cout está definida dentro de la biblioteca iostream.h, que previamente declaramos que íbamos a utilizar. Una función, en este caso main() siempre comienza su ejecución con una instrucción (la que se encuentra en la parte superior), y continúa así hasta que se llegue a la última instrucción (de la parte inferior). Para terminar una instrucción siempre se coloca “;”.

```
return 0;
```

Esta es otra instrucción, en este caso la instrucción return determina que es lo que se devolverá de la función main(). Habíamos declarado que main devolvería un entero, así que la instrucción return devuelve 0. Lo cual a su vez significa que no han ocurrido errores durante su ejecución.

```
}
```

La llave de cierre de la función main() indica el termino del bloque de instrucciones. En algunos programas de ejemplo, notará el uso de dobles diagonales (“//”). Estas diagonales se usan para escribir comentarios de una línea dentro del código del programa. Además podrá encontrar el uso de “/\*” “\*/” estos caracteres encierran un comentario de varias líneas y cualquier cosa que se escriba dentro de ella no influenciará en el desempeño del programa.

También verá que muchas veces utiliza una diagonal invertida (“\”). Este signo se utiliza cuando una instrucción ocupará varias líneas y por razones de espacio en la hoja es mejor dividirla en partes.

## **Proceso de compilación**

La compilación de un programa C o C++ se realiza en varias fases que normalmente son automatizadas y ocultas por los entornos de desarrollo:

1. **Preprocesado** consistente en modificar el código fuente en C o C++ según una serie de instrucciones (denominadas directivas de preprocesado) simplificando de esta forma el trabajo del compilador. Por ejemplo, una de las acciones más



importantes es la modificación de las inclusiones (`#include`) por las declaraciones reales existentes en el fichero indicado.

2. **Compilación** que genera el código objeto a partir del código ya preprocesado.
3. **Enlazado** que une los códigos objeto de los distintos módulos y bibliotecas externas (como las bibliotecas del sistema) para generar el programa ejecutable final.

### Compilación de un programa escrito en C++ en plataforma Linux

g++: Programa compilador de C++

Abrir una consola o línea de comando y escribir

```
g++ ejercicio1.cpp -o ejercicio1
```

**Ejercicio: escriba, compile y ejecute el siguiente programa.**

```
using namespace std;
#include <iostream>
int main()
{
    cout << ""<<endl;
    cout << "Esto es un mensaje"<<endl;
    cout << "Bienvenido al mundo de la programación"<<endl;
    cout << ""<<endl;
    return 0;
}
```

### Actividades de estudios independientes

Revisar los siguientes recursos disponibles en Internet

Taller de lenguaje C Parte 1 ( Operaciones)

<http://es.youtube.com/watch?v=2xUakJkWeSI>

Taller de lenguaje C Parte 2 ( Variables )

<http://es.youtube.com/watch?v=fkOMDxbXfDc>

Taller de lenguaje C Parte 3 ( Control de Flujo - Sentencia If y For)

<http://es.youtube.com/watch?v=Y1NSpOF6LBM>

Taller de lenguaje C Parte 4 ( Control de Flujo - Sentencia If y For)

<http://es.youtube.com/watch?v=fXtwRUTqN-U>



Taller de lenguaje C Parte 5 ( Control de Flujo - Sentencia Switch)

<http://es.youtube.com/watch?v=mUFohxcpE9Q>

Taller de lenguaje C Parte 6 ( Sentencias Iterativas - For)

<http://es.youtube.com/watch?v=54WU3v1e8RM>

Taller de lenguaje C Parte 7 ( Sentencias Iterativas - While)

<http://es.youtube.com/watch?v=DbxIn07vgIk>

Taller de lenguaje C Parte 8 ( Sentencias Iterativas - Do While)

<http://es.youtube.com/watch?v=QYriZlH1bd4>

Taller de lenguaje C Parte 9 ( Sentencias Iterativas - Uso del Break y Continue)

<http://es.youtube.com/watch?v=SJwV2gyGXvQ>

Taller de lenguaje C Parte 10 ( Sentencias Iterativas - Uso de Arreglos)

<http://es.youtube.com/watch?v=xZE7gJvJOIs>

Taller de lenguaje C Parte 11 ( Sentencias Iterativas - Uso de Arreglos)

<http://es.youtube.com/watch?v=j0vkHeu73FA>

Taller de lenguaje C Parte 12 ( Sentencias Iterativas - Uso de Arreglos)

<http://es.youtube.com/watch?v=JMFfR2TepPM>

Taller de lenguaje C Parte 13 ( Sentencias Iterativas - Uso de Arreglos)

<http://es.youtube.com/watch?v=ryAreSuBd2M>

Ejercicio de voltear Palabra

Programación en C (Básico)

<http://es.youtube.com/watch?v=1zF5DqrvAtc>

**Recursos**

<http://sourceforge.net/projects/dev-cpp/>