

Introducción a la programación

El lenguaje de programación C

El lenguaje de programación ANSI C

Componentes del estándar ANSI C

Historia de C

Herramientas de programación en C

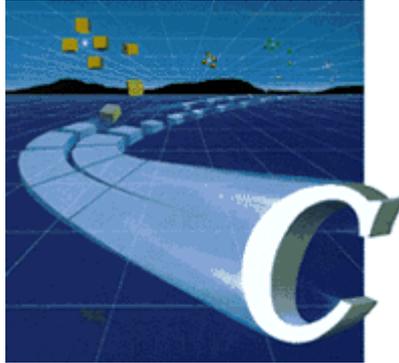
Aplicaciones en C

Programa de ejemplo

Fases en la creación y ejecución de programas en C

Características clave de C

El lenguaje de programación ANSI C



Componentes del estándar ANSI C

- **Compilador**
Traduce a código máquina el código fuente escrito en C
- **Preprocesador**
Actúa sobre el código fuente antes de la compilación
- **Biblioteca estándar**
Conjunto de funciones predefinidas

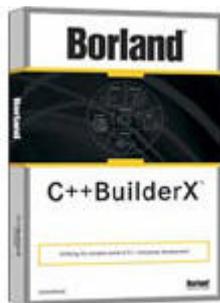
Ejemplos: `stdio.h` E/S básica y ficheros
 `math.h` Funciones matemáticas

Historia de C

Año	
1963	CPL (Combined Programming Language) © Universidades de Cambridge y Londres Simplificación derivada de ALGOL 60 (no implementado)
1967	BCPL (Basic CPL [Combined Programming Language]) © Martin Richards (Universidad de Cambridge) Lenguaje sin tipos creado como herramienta para la construcción de compiladores y sistemas operativos.
1970	B (BCPL simplificado) © Ken Thompson(AT&T Bell Labs, New Jersey) Lenguaje sin tipos creado para el primer sistema UNIX (para la PDP-7 de Digital Equipment Corporation [DEC])
1972	Creación del lenguaje de programación C © Dennis Ritchie (AT&T Bell Labs, New Jersey)
1974	UNIX reescrito en C
1978	Brian W. Kernighan & Dennis Ritchie “The C Programming Language” Prentice Hall, ISBN 0-13-110370-9
1983	C++ © Bjarne Stroustrup (AT&T Bell Labs, New Jersey) Extensión de C (programación orientada a objetos)
1989	Estándar ANSI C #X3J11/99-090 aprobado por el American National Standards Institute
1989	Segunda edición del libro de Kernighan & Ritchie
1995	Java © James Gosling (Sun Microsystems) Similar a C++, aunque más sencillo de aprender y usar
1997	Estándar ANSI C++ aprobado por el American National Standards Institute
2000	C# © Microsoft Corporation Alternativa a Java (base de la plataforma .NET)

Herramientas de programación en C

Existen múltiples compiladores de C
y muchos entornos integrados de desarrollo
[IDEs: Integrated Development Environments]



<http://www.borland.com/>
Borland C++BuilderX



<http://www.eclipse.org/>
Eclipse IDE
Open platform for tool integration



<http://www.microsoft.com/>
Microsoft Visual Studio .NET



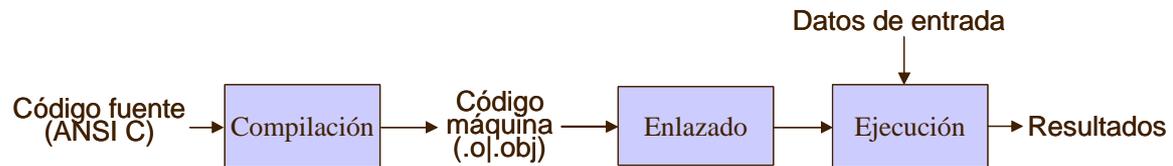
<http://gcc.gnu.org/>
GCC
GNU Compiler Collection



<http://www.openwatcom.org/>
Open Watcom

...

Aplicaciones en C



- Un programa puede tener miles (e incluso millones) de líneas de código fuente. A medida que el programa va creciendo, éste sería inmanejable si no fuésemos dividiéndolo en módulos mucho más pequeños y manejables.
- Todos los programas en C consisten en una o más funciones (fragmentos de código que realizan tareas concretas).
- La ejecución del programa escrito en C comienza en una función denominada `main()`, que siempre debe existir para que podamos crear un programa ejecutable.
- Una aplicación se implementa como un conjunto de funciones a las que podemos llamar desde el programa principal (la función `main`).
- Las distintas funciones de nuestro programa, a su vez, pueden llamar a otras funciones cuando sea necesario.
- La biblioteca estándar de C define algunas funciones comunes que vienen implementadas para que podamos usarlas directamente (sin preocuparnos de su implementación).
- Al compilar un programa en C, creamos el código máquina correspondiente a la parte del programa que hayamos escrito nosotros. A continuación, enlazamos nuestro código máquina con el correspondiente a las funciones ya implementadas en bibliotecas para obtener un programa ejecutable.

Programa de ejemplo

Código fuente en C: Fichero programa.c

```
/* Mi primer programa en C */

#include <stdio.h>

int main (int argc, char *argv)
{
    printf("Hola");

    return 0;
}
```

- La primera línea, entre `/*` y `*/`, es un comentario (algo que no forma parte del código del programa en sí pero que se incluye como aclaración para facilitar su lectura).
- La línea `#include <stdio.h>` le indica al preprocesador de C que incluya en el programa el contenido del fichero de cabecera `stdio.h`, donde están las declaraciones de las funciones estándar de entrada/salida en C.
- La línea `int main(...)` define la cabecera de la función `main`, el punto donde comienza la ejecución de un programa en C.
- Las llaves `{ ... }` sirven para delimitar la secuencia de instrucciones que forman parte de la función `main`.
- La llamada a la función `printf("Hola ");` le indica al ordenador que muestre por pantalla el mensaje que se le indica entre comillas.
- La sentencia `return` indica el valor que devuelve la función `main` (por convención, 0 indica que “todo fue bien”).

Compilación

A continuación tenemos que traducir nuestro programa escrito en C a un programa que pueda ejecutar directamente el ordenador:

```
cc programa.c
```

NOTA: Según el compilador de C que utilicemos, puede que tengamos que escribir `gcc` (GNU C), `cl` (Microsoft), `wcc386` (Watcom), `bcc` o `bcc32` (Borland).

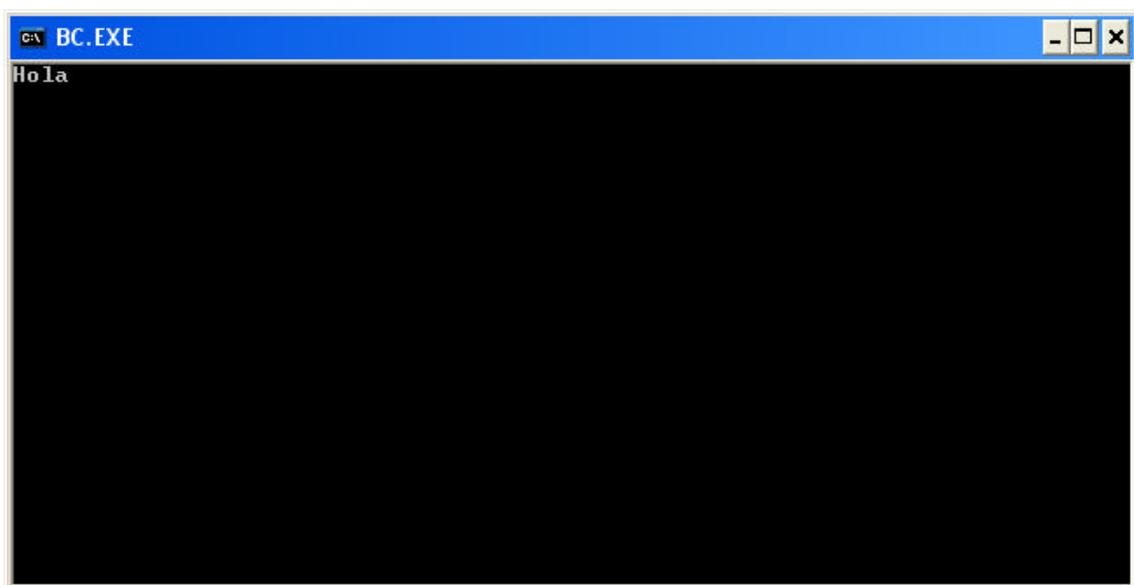
Ejecución

Una vez compilado el programa, para ejecutarlo basta con escribir su nombre

```
programa
```

Resultado

Como resultado de la ejecución, obtenemos lo siguiente:



Fases en la creación y ejecución de programas en C

Fase I: Edición

- Se crea un programa con la ayuda de un editor
- Se almacena en un fichero con extensión .c

Fase II: Preprocesamiento

- El preprocesador realiza sustituciones textuales en el texto del programa escrito en C.

Fase III: Compilación

- El compilador lee el código C tras su preprocesamiento.
- Si se detectan errores sintácticos, el compilador nos informa de ello.
- Se genera el código máquina (en ficheros .o ó .obj)

Fase IV: Enlace

- El enlazador [linker] junta todos los ficheros objeto que sean necesarios para crear el ejecutable (.exe en Windows).

Fase V: Carga

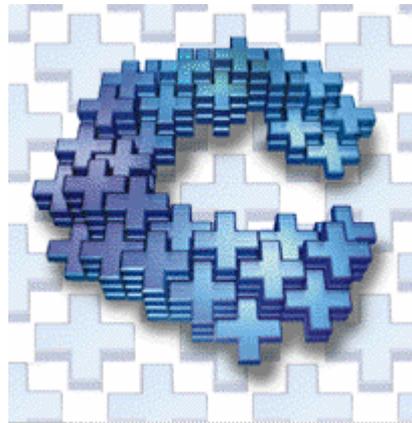
- El sistema operativo lee el fichero ejecutable de disco y lo carga en la memoria principal del ordenador.

Fase VI: Ejecución

- Comienza la ejecución de la función `main` de nuestro programa.

NOTA: Conforme se ejecuta el programa, se hace uso de la memoria principal para almacenar los datos con los que trabaja la aplicación.

Características clave de C



- Lenguaje de programación de propósito general
- Lenguaje estructurado (como Pascal)
- Lenguaje pequeño
- Lenguaje muy eficiente
- Lenguaje portable (independiente de la máquina)
- A menudo es elegido para el desarrollo de software de sistemas: es un lenguaje de programación de alto nivel caracterizado por su acusada proximidad a la máquina
- Origen de muchos de los lenguajes usados actualmente (C++, Java, C#...)

"No es una herramienta para tontos y niños. En manos de un artesano habilidoso es capaz de trabajar de forma potente pero delicada. Su posibilidad de causar un serio daño es tan obvia que ese peligro proporciona el único mecanismo de seguridad; ¡un sano respeto por lo que puede hacer su uso despreocupado!"

*B. Cox, "Software ICs and Objective-C",
Productivity Products International (ahora StepStone Inc.), 1985*