TEMA 3. EL PROCESO DE COMPILACIÓN, DEL CÓDIGO FUENTE AL CÓDIGO MÁQUINA

Programa: Algoritmo (secuencia no ambigua, finita y ordenada de instrucciones para la resolución de un determinado problema) traducido a un lenguaje de programación, de modo que un ordenador es capaz de ejecutarlo.

Programación: Elaboración de un programa de manera que éste sea:

- **Correcto** ⇨ Un programa será correcto ***si hace lo que debe hacer***, de modo que se deben especificar de manera muy clara cuáles son los datos sobre los que se trabajarán y lo que se debe hacer con ellos. Todo debe ser ***documentado y probado*** antes de desarrollarlo.
- **Eficiente** ⇨ Debe consumir la menor cantidad de recursos (tiempo y/o memoria) posible.
- **Claro** ⇨ Es muy importante la claridad y legibilidad de todo programa, ya que facilitará al máximo la tarea de mantenimiento posterior del software.
- **Modular**⇨ Los programas suelen subdividirse en subprogramas (módulos), para reducir la complejidad de aquella parte que se está implementando y facilitar la reutilización de código.

Para la elaboración de un programa hay que distinguir entre las siguientes dos fases:

- **Fase de compilación y linkado (link, montado o enlace)**
- **Fase de ejecución de un programa**

3.1 Fase de compilación y linkado (link, montado o enlace)

Un programa escrito en un lenguaje de alto nivel, no puede ser ejecutado directamente por un ordenador, sino que debe ser traducido a lenguaje máquina.

Las etapas por las que debe pasar un programa escrito en un lenguaje de programación, hasta poder ser ejecutable son:

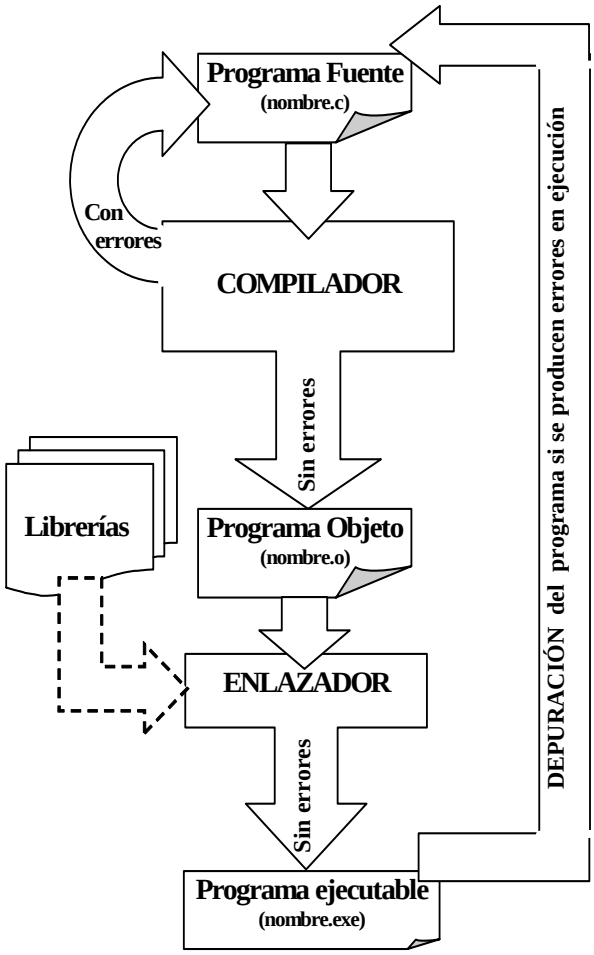

Figura 1. Proceso de transformación de un programa fuente a un programa ejecutable

Programa fuente: Programa escrito en un lenguaje de alto nivel (texto ordinario que contiene las sentencias del programa en un lenguaje de programación). Necesita ser traducido a código máquina para poder ser ejecutado.

Compilador: Programa encargado de traducir los programas fuentes escritos en un lenguaje de alto nivel a lenguaje máquina y de comprobar que las llamadas a las funciones de librería se realizan correctamente.

Programa (o código) objeto: Es el programa fuente traducido (por el compilador) a código máquina. Aún no es directamente ejecutable.

Programa Ejecutable: Traducción completa a código máquina, realizada por el enlazador, del programa fuente y que ya es directamente ejecutable.

Linker (montador o enlazador): Es el programa encargado de insertar al programa objeto el código máquina de las funciones de las librerías (archivos de biblioteca) usadas en el programa y realizar el proceso de montaje, que producirá un **programa ejecutable** .exe. Las ***librerias*** son una colección de código (funciones) ya programado y traducido a código máquina, listo para utilizar en un programa y que facilita la labor del programador.

Como cada lenguaje de programación tiene unas reglas especiales (*sintaxis*) debe existir un compilador específico para cada lenguaje de programación.

Si el programa fuente es sintácticamente **correcto**, el compilador generará el **código objeto**, en caso contrario mostrará una lista con los errores encontrados, no generándose ningún programa objeto, para que procedamos a su *depuración*

Los compiladores emiten mensajes de error o de advertencia durante las fases de compilación, de enlace o de ejecución de un programa:

- Los **errores en tiempo de compilación** son los que se producen antes de la ejecución del programa, durante el proceso de compilación del programa.
- Los **errores en tiempo de ejecución** son los que se producen durante la ejecución del programa. Son los más difíciles de encontrar, no son detectados por el compilador, ya que son errores de lógica, no de sintaxis.

Aunque al compilar un programa no de errores, el programa puede funcionar incorrectamente y/o a dar errores durante su ejecución. Por ejemplo:

- Un programa puede producir resultados erróneos, al equivocarnos (errores lógicos) al programar el algoritmo (sumar en vez de restar, etc.).
- Un programa puede interrumpirse bruscamente, por ejemplo si tenemos que hacer una división y el divisor es cero, etc.

Los errores que se pueden producir en la **fase de compilación** son:

- **Errores fatales:** Son raros. Indican errores internos del compilador. Cuando ocurren la compilación se detiene inmediatamente.
- **Errores de sintaxis:** Son los errores típicos de sintaxis. No detienen la compilación sino que al finalizar ésta se mostrará la lista con todos los errores encontrados. *Algunos errores suelen ser consecuencia de otros cometidos con anterioridad*. Con este tipo de errores no se puede obtener un programa objeto y por lo tanto tampoco el ejecutable.
- **Advertencias o avisos (warnings):** Indican que hay líneas de código sospechosas que a pesar de no infringir ninguna regla sintáctica, el compilador las encuentra susceptibles de provocar un error. Cuando se detecta un warning la compilación no se detiene. Si en un programa fuente sólo se detectan warnings sí que se podrá obtener un programa objeto, que tras el linkado dará lugar a un programa ejecutable.

Con respecto a los **errores en tiempo de ejecución**, encontrar la causa que los provoca es una labor en ocasiones complicada, razón por la cual los EID (Entornos Integrados de Desarrollo, p.ej. DevC++) nos proporcionan una herramienta llamada Depurador que nos ayuda a encontrar los errores lógicos y demás errores producidos en tiempo de ejecución.

Un *depurador (debugger)*, es un programa diseñado específicamente para la detección, verificación y corrección de errores. Los depuradores nos permiten trazar el programa (ejecutarlo sentencia a sentencia) y visualizar el contenido de las variables y direcciones de memoria durante la ejecución del programa. Además permiten alterar el flujo de ejecución del mismo, cambiar los valores de las variables e introducir puntos de parada.

Pasos para la elaboración y ejecución de un programa:

Los pasos a seguir los podemos resumir de la siguiente manera:

1 º. Escribir el código fuente, por ejemplo con el editor del EID.

2 º. Compilar el fichero fuente

3 º. Si se producen errores de sintaxis (o warnings) volver al editor y eliminar los errores de sintaxis.

4 º. Si no hay errores se obtendrá el código objeto y el enlazador construirá el archivo ejecutable.

5 º. Una vez tengamos el archivo ejecutable, será el sistema operativo el encargado de colocar el programa en la memoria central y ejecutarlo.

6 º. Comprobar el funcionamiento del programa.

7 º. Si se detecta errores o un mal funcionamiento del programa, activar el depurador para trazar el programa y ejecutarlo sentencia a sentencia.

8 º. Una vez que hayamos encontrado la causa del error, volveremos al editor y lo corregimos.

9 º. El proceso de compilar, enlazar y ejecutar el programa lo repetiremos hasta que no se produzcan errores.

3.2 Fase de ejecución de un programa

Una vez que tenemos el programa en lenguaje máquina, para poderlo ejecutar hay que introducirlo en la memoria.

1. Una utilidad del S.O. llamada ***cargador*** colocará el programa, y sus datos de entrada, en memoria principal, preparándolo para su ejecución.
2. El S.O. le pasa el control a la C.P.U. para que comience la ejecución del programa, realizando la Unidad de Control los siguientes pasos (fases):
   - ⇨ ***Captación de la instrucción***: Lee de la Memoria Principal la instrucción a ejecutar.
   - ⇨ ***Ejecución de la instrucción***: Interpreta la instrucción leída y envía señales de control a las unidades que deban intervenir en su ejecución. Tras dicha ejecución se establece cuál será la siguiente instrucción a ejecutar.

3.3 El compilador de C. Características generales.

- El lenguaje C es un lenguaje muy **potente y eficiente** de ***nivel medio***: combina elementos de lenguajes de alto nivel con la funcionalidad del lenguaje ensamblador. Es adecuado para la programación de sistemas.
- Es un lenguaje de propósito general (puede utilizarse para desarrollar sistemas operativos, gestores de bases de datos, etc).
- El código de C es muy **portable**: se puede adaptar el software escrito para un tipo de computadora a otra computadora sin hacer muchos cambios.
- El lenguaje C es un ***lenguaje estructurado***: permite seccionar y esconder (mediante subrutinas y variables locales) del resto del programa toda la información e instrucciones necesarias para realizar una determinada tarea.
- El lenguaje C **sólo tiene 32 palabras clave**, (BASIC por ejemplo tiene 159).
- Suele incluir potentes librerías de funciones que aumentan su potencia.
- El lenguaje C es ***compilado, no interpretado***. Un intérprete lee el código fuente de un programa línea a línea y las traduce online para que la CPU las pueda interpretar, pero no genera ningún programa objeto. Un compilador lee el programa entero, lo traduce y genera un *código objeto* directamente ejecutable y entendible por el microprocesador.