TEMA 2:

LENGUAJES DE PROGRAMACIÓN

María Aparicio Gil
Ricardo Blanco Jiménez
Alejandro Pérez-Moneo Nieto
Cristina Villoria Valiente
Alejandro Zazo Mendo

ÍNDICE

TEMA 2: LENGUAJES DE PROGRAMACION

COTENIDOS

- Concepto de lenguaje de programación
- Historia y evolución de los lenguajes de programación
- Paradigma de programación
- Algunas clasificaciones de lenguajes de programación
- Traductores

CONCEPTOS CLAVE

- Compilación
- Compilador
- Intérprete
- Lenguaje de programación
- Lenguaje ensamblador
- Lenguaje maquina

AL FINAL DEL TEMA SABRAS

- Definición de lenguaje de programación
- Conocer la evolución de los lenguajes de programación desde el lenguaje maquina hasta los lenguajes de alto nivel
- Distinguir entre los tipos de lenguajes que existen dependiendo de su enfoque
- La historia de los herederos de C

1.1 CONCEPTO DE LENGUAJE DE PROGRAMACIÓN

Los lenguajes de programación se utilizan para escribir programas. Los programas de las computadoras modernas constan de secuencias de instrucciones que se codifican como secuencias de dígitos numéricos que podrán entender dichas computadoras.

El sistema de codificación se conoce como lenguaje máquina que es el lenguaje nativo de una computadora.

Cada lenguaje de programación tiene un conjunto de instrucciones (acciones u operaciones que debe realizar la maquina) que la computadora podrá entender directamente en su código maquina o bien se traducirán a dicho código máquina. Las instrucciones básicas y comunes en casi todos los lenguajes de programación son:

- *Instrucciones de entrada/salida:* instrucciones de trasferencia de información entre dispositivos periféricos y la memoria central, tales como "leer de..." o bien "escribir en..."
- *Instrucciones de cálculo:* instrucciones para que a computadora pueda realizar operaciones aritméticas.
- *Instrucciones de control:* instrucciones que modifican la secuencia de la ejecución del programa.

Además de estas instrucciones y dependiendo del procesador y del lenguaje de programación existirán otras que conformaran el conjunto de instrucciones y junto con las reglas de sintaxis permitirían escribir los programas de las computadoras. Los principales tipos de lenguajes de programación son:

- Lenguajes máquina
- Lenguajes de bajo nivel
- Lenguajes de alto nivel

1.2 EVOLUCIÓN DE LOS LENGUAJES DE PROGRAMACIÓN

En la década de los 40 cuando nacían las primeras computadoras digitales el lenguaje que se utilizaba para programar era el lenguaje maquina que se traducían directamente el código maquina (Código binario) comprensible por las computadoras. Las instrucciones en lenguaje máquina dependerán de cada computadora y debido a la dificultad de su escritura, los investigadores de la época simplificaron el proceso de programación desarrollando sistemas de notación en los cuales las instrucciones se representaban en formatos nemónicos (nemotécnicos) en de en formatos numéricos que eran más difíciles de recordar.

Para convertir programas escritos en código nemotécnicos a lenguaje maquina, se desarrollaron programas ensambladores. Es decir, los ensambladores son programas que traducen otros programas escritos en código nemotécnico en instrucciones numéricas en lenguaje maquina que son compatibles y legibles por la maquina. Estos programas de traducción se llaman ensambladores porque su tarea es ensamblar las instrucciones reales de la maquina con los nemotécnicos e identificadores que representan las instrucciones escritas en ensamblador. A estos lenguajes se les denominó de segunda generación, reservando el nombre de primera generación para los lenguajes maquina.

En la década de los 50 y 60 comenzaron a desarrollarse lenguajes de programación de tercera generación que diferían de las generaciones anteriores en que sus instrucciones o primitivas eran de alto nivel e independientes de la maquina. Estos lenguajes se llamaron lenguajes de alto nivel. Los mas conocidos son FORTRAN, COBOL, Pascal, BASIC, C, C++, Ada, Java, C#, HTML, XML,….

Ejemplo de programa en FORTRAN

```
Program Ecua_linales2
implicit none
real*4 x,y,a1,b1,a2,b2,c1,c2
write(*,*)'Este programa resuelve un
sistema de ecuaciones de la forma'
write(*,*)'a1*X+b1*Y=c1'
write(*,*)'a2*X+b2*Y=c2'
write(*,*)'Las soluciones
son:','X=',x,'Y=',y
end if
end program
```

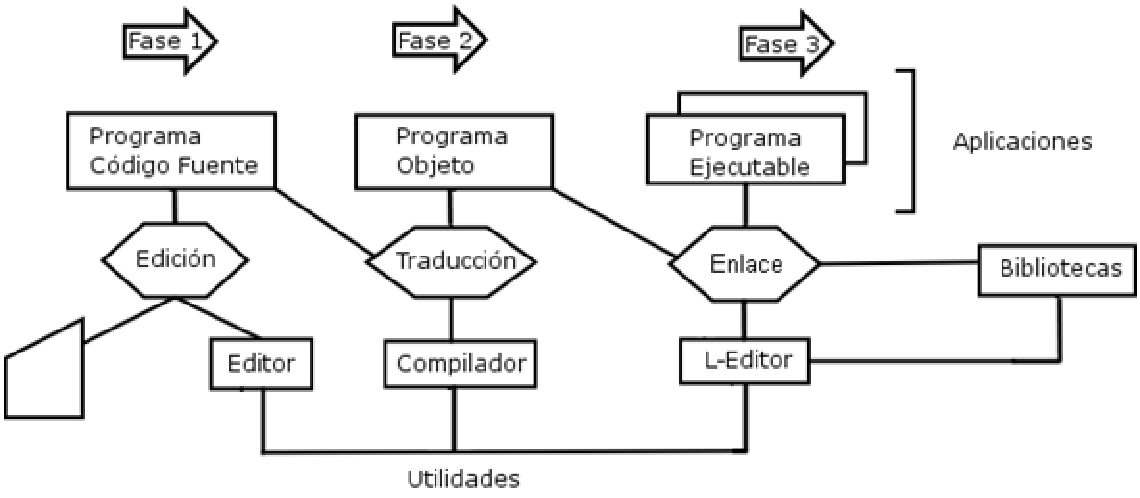
Ejemplo de programa en COBOL

```
000048 01  verifica-bisiesto.
000049     03  entero     pic 9(4) value
0.
000050     03  resto      pic 9(4) value
0.
000051
000052       screen section.
000053       01  pantallas.
000054       02  pantalla0.
000055        03 line 1 column 1 blank
screen.
000056        03 line 5 column 20 value
"programa de fechas".
```

Los lenguajes de programación de alto nivel se componen de un conjunto de instrucciones o primitivas más fáciles de escribir y recordar su función que los lenguajes maquina y ensamblador. Sin embargo, necesitan los programas escritos en un lenguaje de alto nivel, ser traducidos a código maquina; para ello se requiere e un programa denominado traductor. Estos programas de traducción son conocidos técnicamente, compiladores. De este modo existen compiladores de C, FORTRAN, Pascal etc.

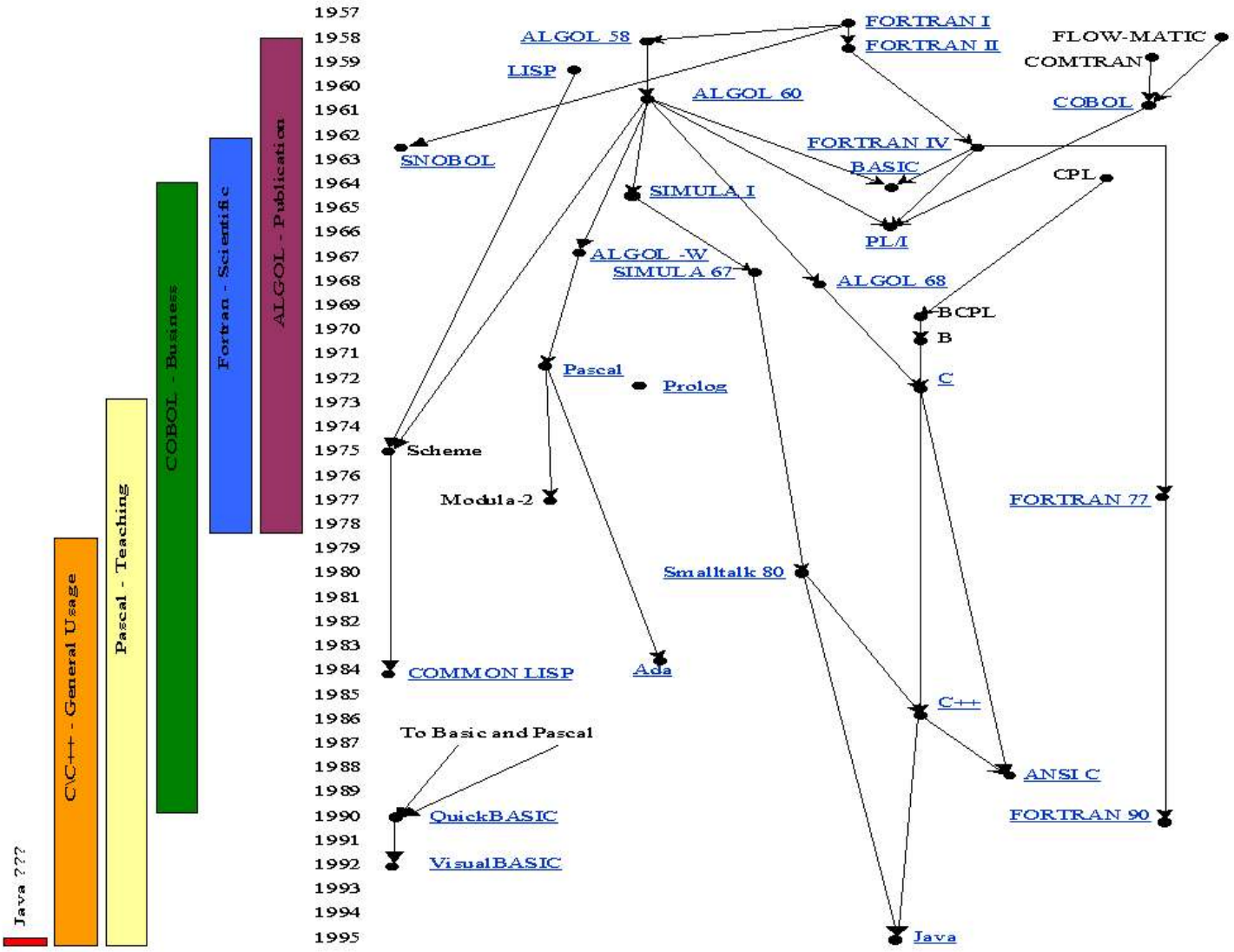
También surgió una alternativa a los traductores compiladores como medio de implementación de lenguajes de tercera generación y se denominaron intérpretes. Estos programas eran similares a los traductores excepto que ellos ejecutaban las instrucciones a medida que se traducían, en lugar de guardar la versión completa traducida para su uso posterior. Es decir, en vez de producir una copia de un programa en lenguaje maquina que se ejecuta más tarde (este es el caso de la mayoría de los lenguajes), un interprete ejecuta realmente un programa desde su formato de alto nivel, instrucción a instrucción. Cada tipo de traductor tiene sus ventajas e inconvenientes, aunque hoy en día prácticamente los traductores utilizados son casi todos compiladores pos su mayor eficiencia y rendimiento.

Sin embargo en el aprendizaje de programación se suele comenzar también su aprendizaje con el uso de los lenguajes algorítmicos, similares a los lenguajes naturales, mediante instrucciones escritas en pseudocódigo que son palabras o abreviaturas de palabras escritas en ingles, español, portugués, etc. posteriormente se realiza la conversión al lenguaje de alto nivel que se vaya a utilizar realmente en la computadora, tal como C, C++ o Java. Esta técnica facilita la escritura de algoritmos como paso previo a la programación.



1.3 PARADIGMA DE PROGRAMACIÓN

La evolución de los lenguajes de programación ha ido paralela a la idea de paradigma de programación: enfoques alternativos a los procesos de programación. En realidad un paradigma de programación representa fundamentalmente enfoques diferentes para la construcción de soluciones a problemas y por consiguiente afectan al proceso completo de desarrollo de software. Los paradigmas de programación clásicos son: procedimental, funcional, declarativo y orientado a objetos.



1.3.1 Lenguajes imperativos (procedímentales)

El paradigma imperativo o prodecimental representa el enfoque o método tradicional de programación. Un lenguaje imperativo es un conjunto de instrucciones que se ejecutan una pos una, de principio a fin, de modo secuencial excepto cuando intervienen instrucciones de salto de secuencia o control. Este paradigma define el proceso de programación como el desarrollo de una secuencia de órdenes que manipulan los datos para producir los resultados deseados. Por consiguiente, el paradigma imperativo sea un enfoque del proceso de programación mediante la realización de un algoritmo que resuelve de modo manual el problema y la continuación expresa ese algoritmo como una secuencia de órdenes. En un lenguaje procedimental cada instrucción es una orden u órdenes para que la computadora realice alguna tarea especifica.

Los lenguajes de programación prodecimentales por excelencia son FORTRAN, COBOL, Pascal, BASIC, ALGOL, C y Ada.

1.3.2 Lenguajes declarativos

En contraste con el paradigma imperativo en paradigma declarativo solicita al programador que describa el problema en lugar de encontrar una solución algorítmica al problema, es decir, un lenguaje declarativo utiliza el principio del razonamiento lógico para responder a las preguntas o cuestiones consultadas. Se base en la lógica formal y en el cálculo de predicados de primer orden. El razonamiento lógico se basa en la deducción. El lenguaje declarativo por excelencia es el Prolog.

1.3.3 Lenguajes orientados a objetos

El paradigma orientado a objetos se asocia con el proceso de programación llamado programación orientada a objetos (POO) consistente en un enfoque totalmente distinto al proceso procedimental. El enfoque orientado a objetos guarda analogía con la vida real. El desarrollo del software se basa en el diseño y construcción de objetos que se componen a su vez de datos y operaciones que esos datos. El programador define en primer lugar los objetos del problema y a continuación los datos y operaciones que actuaran sobre esos datos. Las ventajas de la programación orientada a objetos se derivan esencialmente de la estructura modular existente en la vida real y el modo de respuesta de estos módulos u objetos a mensajes o eventos que se producen en cualquier instante.

Los orígenes de la POO se remontan a los tipos abstractos de datos como parte constitutiva de una estructura de datos.

C++ lenguaje orientado a objetos, por excelencia, es una extensión del lenguaje C y contiene las tres propiedades más importantes: encapsulamiento, herencia y polimorfismo.

Hoy en día Java y C# son herederos directos de C++ y C, y constituyen los lenguajes orientados a objetos más utilizados en la industria del software.

1.4 ALGUNAS CLASIFICACIONES DE LOS LENGUAJES DE PROGRAMACIÓN

Los principales tipos de lenguajes utilizados en la actualidad son tres:

- Lenguaje máquina
- Lenguaje de bajo nivel (ensamblador)
- Lenguaje de alto nivel

1.4.1 Lenguaje máquina

Los lenguajes máquina son aquellos que están escritos en lenguaje directamente inteligible por la máquina (computadora), ya que sus instrucciones son cadenas binarias (cadenas o series de caracteres, dígitos, 0 y 1) que especifican una operación, y las posiciones (dirección) de memorias implicadas en la operación se denominan instrucciones de máquinas o código máquina. El código máquina es el conocido código binario.

Las instrucciones en lenguaje máquina dependen del hardware de la computadora y, por tanto, diferirán de una computadora a otra. El lenguaje máquina de un PC será diferente de un sistema HP, Dell, Compaq o un sistema de IBM.

Las ***ventajas*** de programar en lenguaje máquina se refieren, fundamentalmente, a la posibilidad de cargar (transferir un programa a la memoria) sin necesidad de traducción posterior, lo que supone una velocidad de ejecución superior a cualquier otro lenguaje de programación.

Los ***inconvenientes***, en la actualidad, superan las ventajas, lo que hace prácticamente no recomendables los lenguajes máquina al programador de aplicaciones. Estos inconvenientes son:

- dificultad y lentitud en la codificación
- poca fiabilidad
- dificultad grande de verificar y poner a punto los programas
- los programas sólo son ejecutables en el mismo procesador (UPC, Unidad Central de Proceso)

Para evitar los lenguajes máquina, desde el punto de vista del usuario, se han creado otros lenguajes que permiten escribir programas con instrucciones similares al lenguaje humano (casi siempre inglés, aunque existen excepciones como es el caso de las antiguas versiones españolas del lenguaje LOGO). Estos lenguajes de denominan de alto y bajo nivel.

1.4.2 Lenguaje de bajo nivel

Los lenguajes de bajo nivel son más fáciles de utilizar que los lenguajes máquina, pero, al igual que ellos, dependen de la máquina en particular. El lenguaje de bajo nivel por excelencia es el ensamblador. Las instrucciones en lenguaje ensamblador son instrucciones conocidas como nemotécnicas. Por ejemplo, nemotécnicos típicos de operaciones aritméticas son: en inglés, ADD, SUB, DIV, etc.; en español, SUM, RES, DIV, etc.

Una instrucción típica de suma sería: ADD, M, N, P

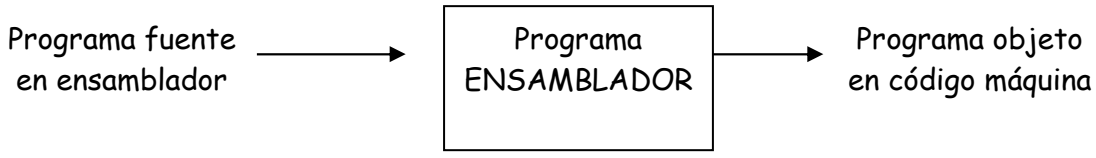
Esta instrucción podía significar "sumar el contenido en la posición de memoria M al número almacenado en la posición de memoria N y situar el resultado en la posición de memoria P". Evidentemente, es mucho más sencillo recordar la instrucción anterior con un nemotécnico que su equivalente en código máquina: 0110 1001 1010 1011

Un programa escrito en lenguaje ensamblador no puede ser ejecutado directamente por la computadora, en esto se diferencia esencialmente del lenguaje máquina, sino que requiere una fase de traducción al lenguaje máquina.

El lenguaje original escrito en lenguaje ensamblador se denomina programa fuente y el programa traducido en lenguaje máquina se conoce como programa objeto ya directamente inteligible por la computadora.

El traductor de programas fuente a objeto es un programa llamado ensamblador, existente en casi todas las computadoras.

No se debe confundir el programa ensamblador, encargado de efectuar la traducción del programa fuente escrito a lenguaje máquina, con el lenguaje ensamblador, lenguaje de programación con una estructura y gramática definidas.



Los lenguajes ensambladores presentan la ventaja frente a los lenguajes máquina de su mayor facilidad de codificación y, en general, su velocidad de cálculo.

Los *inconvenientes* más notables de los lenguajes ensambladores son:

- Dependencia total de la máquina, lo que impide la transportabilidad de los programas (posibilidad de ejecutar un programa en diferentes máquinas).
- La formación de los programadores es más compleja que la correspondiente a los programadores de alto nivel, ya que exige no sólo técnicas de programación, sino también el conocimiento del interior de la máquina.

Hoy día los lenguajes ensambladores tienen sus aplicaciones muy reducidas en la programación de aplicaciones y se centran en aplicaciones de tipo real, control de procesos y de dispositivos electrónicos, etc.

1.4.3 Lenguaje de alto nivel

Los lenguajes de alto nivel son los más utilizados por los programadores. Están diseñados para que las personas escriban y entiendan los programas de un modo mucho más fácil que los lenguajes máquina y ensambladores. Otra razón es que un programa escrito en lenguaje de alto nivel es independiente de la máquina; esto es, las instrucciones del programa de la computadora no dependes del diseño del hardware o de una computadora en particular, lo que significa la posibilidad de poder ser ejecutados con poca o ninguna modificación en diferentes tipos de computadoras; al contrario que los programas en lenguaje máquina o ensamblador, que sólo se pueden ejecutar en un determinado tipo de computadora.

Los lenguajes de alto nivel presentan las siguientes *ventajas:*

- El tiempo de formación de los programadores es relativamente corto comparado con otros lenguajes
- La escritura de programas se basa en reglas sintácticas similares a los lenguajes humanos. Nombres de instrucciones puede ser READ, WRITE, PRINT, OPEN, etc.
- Las modificaciones y puestas a punto de los programas son más fáciles.
- Reducción de coste de los programas
- Transportabilidad.

Los *inconvenientes* se concretan en:

- Incremento del tiempo de puesta a punto, al necesitarse diferentes traducciones del programa fuente para conseguir el programa definitivo.
- No se aprovechan los recursos internos de la máquina, que se explotan mucho mejor en lenguajes máquinas y ensambladores.
- Aumento de la ocupación de memoria
- El tiempo de ejecución de los programas es mucho mayor.

Al igual que sucede con los lenguajes ensambladores, los programa fuente tienen que ser traducidos por lo llamados traductores, llamados en este caso compiladores e intérpretes.

Los lenguajes de programación de alto nivel existentes hoy son numerosos, aunque la práctica demuestra que su uso mayoritario se reduce a:

- **C**
- **C++**
- **COBOL**
- **FORTRAM**
- **PASCAL**
- **VISUAL**
- **BASIC**
- **JAVA**
- **C#**

Están muy extendidos:

- **Ada-95**
- **Modula-2**
- **Prolog**
- **LISP**
- **Smalltalk**
- **Eiffel**

Son de gran uso en el mundo profesional:

- **Borland**
- **Delphi**
- **SQL**
- **Power**
- **Builder**

Aunque hoy en día el mundo Internet consume gran cantidad de recursos en forma de lenguajes de programación, tales como **Java, HTML, XML, JavaScript, Visual** y últimamente **C#** y **PHP**.

1.5 TRADUCTORES

Los traductores de lenguaje son programas que traducen a su vez los programas fuente escritos en lenguajes de alto nivel a código máquina. Los traductores se dividen en:

- Ensambladores
- Intérpretes
- Compiladores

1.5.1 Ensambladores

Son los encargados de transformar o traducir directamente los programas escritos en ensamblador a su equivalente en código máquina o binario para que puedan ser ejecutados por la CPU.

1.5.2 Intérpretes

Un intérprete es un traductor que toma un programa fuente, lo traduce y a continuación lo ejecuta. Los programas intérpretes clásicos, como BASIC, prácticamente ya no se utilizan, aunque puede encontrar una vieja computadora que funcione con la versión QBASIC bajo el sistema operativo DOS que corre en las computadoras personales. Sin embargo, está muy extendida la versión interpretada del lenguaje Smalltalk, un lenguaje orientado a objetos puro. Los intérpretes han vuelto a renacer con la aparición de Java, ya que para entender el código en bytes al que traduce un compilador Java necesita un intérprete.

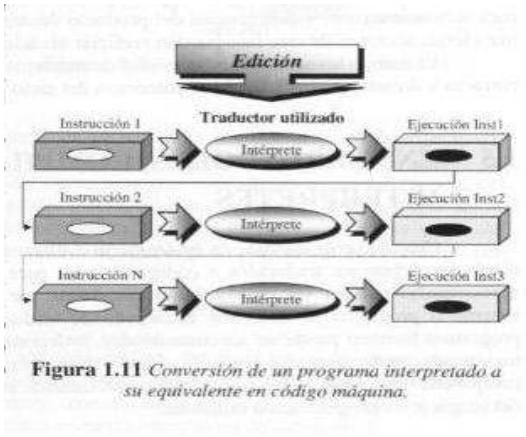


Figura 1.11 *Conversión de un programa interpretado a su equivalente en código máquina.*

1.5.3 Compiladores

Después que se ha diseñado el algoritmo y escrito el programa en un papel, se debe comenzar el proceso de introducir el programa en un archivo (fichero) en el disco duro de la computadora. La introducción y modificación de su programa en un archivo se hace utilizando un editor de texto o simplemente un editor, un programa que viene con su computadora, normalmente, y que le permite el almacenamiento y recuperación de lo que se ha escrito en disco.

El programa que se ha introducido está escrito en C o en Java, pero ni C ni Java son lenguajes máquina de la computadora, muy al contrario con lenguajes de alto nivel diseñados para hace más fácil la programación que utilizando el lenguaje máquina. Una computadora no entiende los lenguajes de alto nivel. En consecuencia, un programa escrito en un lenguaje de alto nivel debe ser traducido a un lenguaje que la máquina pueda comprender. Los lenguajes que la computadora puede comprender se llaman lenguajes de bajo nivel. La traducción de un programa escrito en un lenguaje de alto nivel, como C++ o Java, aun lenguaje que pueda entender la computadora es hace mediante otro programa conocido como **compilador**.

Cuando se ejecuta un programa en lenguaje de alto nivel, se está ejecutando realmente una traducción de ese programa a un lenguaje de bajo nivel. Por consiguiente antes de que se ejecute un programa escrito en un lenguaje de alto nivel, se debe ejecutar en primer lugar el compilador en el programa. Cuando se ejecuta un compilador sobre su programa se dice que se **compila** el programa.

El programa escrito en un lenguaje de programación se denomina programa fuente o código fuente. El programa traducido a lenguaje de bajo nivel producido por el compilador se denomina normalmente el programa objeto o código objeto.

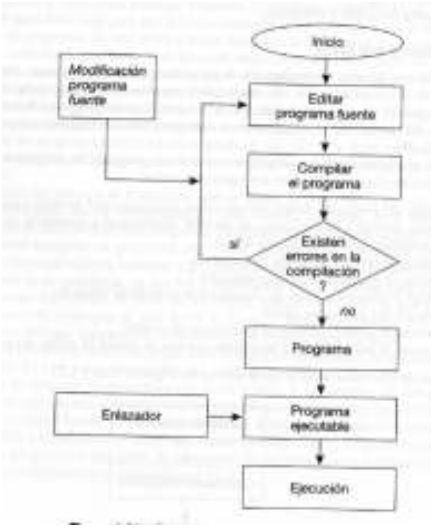
1.5.2.1 La compilación y sus fases

La compilación es el proceso de traducción de programas fuente a programas objetos. El programa objeto obtenido de la compilación ha sido traducido normalmente a código máquina.

Para conseguir el programa máquina real se debe utilizar un programa llamado *montador o enlazador*. El proceso de montaje conduce a un programa en lenguaje máquina directamente ejecutable.

El proceso de ejecución de un programa escrito en un lenguaje de programación y mediante un compilador suele tener los siguientes pasos:

1. Escritura del programa fuente con u editor y guardarlo en un dispositivo de almacenamiento
2. Introducir el programa fuente en memoria.
3. Compilar el programa con el compilador C.
4. Verificar y corregir errores de compilación.
5. Obtención del programa objeto.
6. El enlazador obtiene el programa ejecutable.
7. Se ejecuta el programa y, si no existen errores, se tendrá la salida del programa.



RESUMEN

- Los programas escritos en un lenguaje de alto nivel deben ser traducidos por un compilador antes de que se puedan ejecutar en una maquina especifica. En la mayoría de los lenguajes de programación se requiere un compilador para cada maquina en la que se desea ejecutar a programas escritos en un lenguaje especifico.

- Los lenguajes de programación se clasifican en:
  - Alto nivel: Pascal, FORTRAN, Visual Basic, C, Ada, Modula-2, C++, Java, Delphi, C#, etc.
  - Bajo nivel: ensamblador
  - Maquina: código máquina
  - Diseño de Web: SMGL, HTML, XML, PHP…

- Los programas traductores de lenguajes son:
  - Compiladores
  - Intérpretes

- C es un lenguaje de programación que contiene excelentes características como lenguaje de aprendizaje de programación y lenguaje profesional de propósito general: básicamente es un entorno de programación con editor y compilador incorporado

BIBLIOGRAFÍA

- Luís Joyanes Aguilar, Ignacio Zahonero Martínez. "**Programación en c: metodología, algoritmos y estructuras de datos**". Mc Graw Hill segunda edición, 2005
- Francisco José García Peñalvo, Juan Andrés Hernández Simón, Roberto Theron Sánchez, Vivian López Batista, Iván Álvarez Navia. "**Programación en c**". Departamento de Informática y Automática Facultad de Ciencias Universidad de Salamanca, 2004
- José López Herranz y Enrique Quero Catalinas. "**Fundamentos de Programación**". Paraninfo, 2003
- Luís Joyanes Aguilar. "**Fundamentos de Programación, Algoritmos, Estructuras de datos y Objetos**". Mc Graw Hill, 2003