

# Programación

---

La **programación** es el proceso utilizado para idear y ordenar las acciones necesarias para realizar un proyecto, preparar ciertas máquinas o aparatos para que empiecen a funcionar en el momento y en la forma deseados o elaborar programas para su empleo en computadoras.<sup>1</sup>

En la actualidad, la noción de programación se encuentra muy asociada a la creación de aplicaciones de informática y videojuegos. En este sentido, es el proceso por el cual una persona desarrolla un programa, valiéndose de una herramienta que le permita escribir el código (el cual puede estar en uno o varios lenguajes, como C++, Java y Python, entre otros) y de otra que sea capaz de “traducirlo” a lo que se conoce como lenguaje de máquina, que puede “comprender” el microprocesador.<sup>2</sup>

## Índice

---

**Funcionamiento de un programa**

**Léxico y programación**

**Programas y algoritmos**

**Compilación**

**Programación e ingeniería del software**

**Referencias históricas**

**Objetivos de la programación**

**Ciclo de vida del software**

**Véase también**

**Referencias**

**Enlaces externos**

## Funcionamiento de un programa

---

Para crear un programa, y que la computadora lo interprete y ejecute, las instrucciones deben escribirse en un lenguaje de programación. En sus comienzos las computadoras interpretaban solo instrucciones en un lenguaje específico, del más bajo nivel, conocido como código máquina, siendo éste excesivamente complicado para programar. De hecho solo consiste en cadenas de números 1 y 0 (sistema binario). Para facilitar el trabajo de programación, los primeros científicos, que trabajaban en el área, decidieron reemplazar las instrucciones, secuencias de unos y ceros, por mnemónicos, que son abreviaturas en inglés de la función que cumple la instrucción; las codificaron y crearon así un lenguaje de mayor nivel, que se conoce como Assembly o lenguaje ensamblador. Por ejemplo, para sumar se podría usar la letra A de la palabra inglesa *add* (sumar). En realidad escribir en lenguaje ensamblador es básicamente lo mismo que hacerlo en lenguaje máquina, pero las letras y palabras son bastante más fáciles de recordar y entender que secuencias de números binarios. A medida que la complejidad de las tareas que realizaban las computadoras aumentaba, se hizo necesario disponer de un método más sencillo para programar. Entonces, se crearon los lenguajes de alto nivel. Mientras que una tarea tan trivial como multiplicar dos números puede necesitar un conjunto de instrucciones en lenguaje ensamblador, en un lenguaje de alto nivel bastará con solo una. Una vez que se termina de escribir un programa, sea en ensamblador o en algunos casos, lenguajes de alto nivel, es necesario compilarlo, es decir,

traducirlo completo a lenguaje máquina.<sup>3</sup> Eventualmente será necesaria otra fase denominada comúnmente link o enlace, durante la cual se anexan al código, generado durante la compilación, los recursos necesarios de alguna biblioteca. En algunos lenguajes de programación, puede no ser requerido el proceso de compilación y enlace, ya que se puede trabajar en modo intérprete. Esta modalidad de trabajo es equivalente pero se realiza instrucción por instrucción, se traduce a medida que es ejecutado el programa.

## Léxico y programación

---

La programación se rige por reglas y un conjunto más o menos reducido de órdenes, expresiones, instrucciones y comandos que tienden a asemejarse a una lengua natural acotada (en inglés); y que además tienen la particularidad de una reducida ambigüedad. Cuanto menos ambiguo es un lenguaje de programación se dice que es más potente. Bajo esta premisa, y en el extremo, el lenguaje más potente existente es el binario, con ambigüedad nula (lo cual lleva a pensar así del lenguaje ensamblador).

En los lenguajes de programación de alto nivel se distinguen diversos elementos entre los que se incluyen el léxico propio del lenguaje y las reglas semánticas y sintácticas. Estos lenguajes tienen la particularidad de utilizar palabras del idioma inglés dentro de su léxico, por lo que muchas de ellas tienen una función específica dentro del lenguaje con el que se está trabajando y no se las puede utilizar de manera diferente, son las denominadas palabras reservadas. Por otro lado, otra particularidad de los lenguajes de programación de alto nivel y sus herramientas de desarrollo es el permitir a los programadores el uso de comentarios (frases o párrafos sin funcionalidad en el programa), a fin de que otros programadores entiendan más fácilmente la funcionalidad del código creado.<sup>4</sup>

## Programas y algoritmos

---

Un algoritmo es una secuencia no ambigua, finita y ordenada de instrucciones que han de seguirse para resolver un determinado problema. Un programa normalmente implementa y contiene uno o más algoritmos. Un algoritmo puede expresarse de distintas maneras: en forma gráfica, como un diagrama de flujo, en forma de código como en pseudocódigo o un lenguaje de programación, en forma explicativa.

Los programas suelen subdividirse en partes menores, llamadas módulos, de modo que la complejidad algorítmica de cada una de las partes sea menor que la del programa completo, lo cual ayuda a simplificar el desarrollo del programa. Esta es una práctica muy utilizada y se conoce como "refino progresivo".

Según Niklaus Wirth, un programa está formado por los algoritmos y la estructura de datos.

La programación puede seguir muchos enfoques, o paradigmas, es decir, diversas maneras de formular la resolución de un problema dado. Algunos de los principales paradigmas de programación son:

- Programación declarativa
- Programación imperativa
- Programación estructurada
- Programación modular
- Programación orientada a objetos
- Programación orientada a eventos

## Compilación

---

El programa escrito en un lenguaje de programación de alto nivel (fácilmente comprensible por el programador) es llamado *programa fuente* y no se puede ejecutar directamente en una computadora. La opción más común es compilar el programa obteniendo un módulo objeto, aunque también, si el lenguaje lo soporta, puede ejecutarse en forma directa pero solo a través de un intérprete. Algunos lenguajes, tal como BASIC, disponen de ambas formas de ejecución, lo cual facilita la tarea de depuración y prueba del programa.

El código fuente del programa se debe someter a un proceso de traducción para convertirlo a lenguaje máquina o bien a un código intermedio, generando así un módulo denominado "objeto". A este proceso se le llama compilación.

Habitualmente la creación de un programa ejecutable (un típico .exe para Microsoft Windows o DOS) conlleva dos pasos: el primer paso se llama compilación (propiamente dicho) y traduce el código fuente, escrito en un lenguaje de programación y almacenado en un archivo de texto, a código en bajo nivel (normalmente a código objeto, no directamente a lenguaje máquina). El segundo paso se llama enlazado en el cual se enlaza el código de bajo nivel generado de todos los ficheros y subprogramas que se han mandado compilar y se añade el código de las funciones necesarias que residen en bibliotecas externas, para que el ejecutable pueda comunicarse directamente con el sistema operativo, traduciendo así finalmente el código objeto a código máquina, y generando un módulo ejecutable.

Estos dos pasos se pueden hacer por separado, almacenando el resultado de la fase de compilación en archivos objetos (un típico .o para Unix, .obj para MS-Windows y DOS); para enlazarlos en fases posteriores, o crear directamente el ejecutable; con lo que la fase de compilación puede almacenarse de forma temporal. Un programa podría tener partes escritas en varios lenguajes, por ejemplo, Java, C, C++ y ensamblador, que se podrían compilar de forma independiente y luego enlazar juntas para formar un único módulo ejecutable.

## **Programación e ingeniería del software**

---

Existe una tendencia a identificar el proceso de creación de un programa informático con la programación, que es cierta cuando se trata de programas pequeños para uso personal, y que dista de la realidad cuando se trata de grandes proyectos.

El proceso de creación de software, desde el punto de vista de la ingeniería, incluye mínimamente los siguientes pasos:

1. Reconocer la necesidad de un programa para solucionar un problema o identificar la posibilidad de automatización de una tarea.
2. Recolectar los requisitos del programa. Debe quedar claro qué es lo que debe hacer el programa y para qué se necesita.
3. Realizar el análisis de los requisitos del programa. Debe quedar claro *qué* tareas debe realizar el programa. Las pruebas que comprueben la validez del programa se pueden especificar en esta fase.
4. Diseñar la arquitectura del programa. Se debe descomponer el programa en partes de complejidad abordable.
5. Implementar el programa. Consiste en realizar un diseño detallado, especificando completamente todo el funcionamiento del programa, tras lo cual la codificación (programación propiamente dicha) debería resultar inmediata.
6. Probar el programa. Comprobar que pasan pruebas que se han definido en el análisis de requisitos.
7. Implantar (instalar) el programa. Consiste en poner el programa en funcionamiento junto con los componentes que sean necesarios (bases de datos, redes de comunicaciones, etc.).

La ingeniería del software se centra en los pasos de planificación y diseño del programa, mientras que antiguamente (programación artesanal) la realización de un programa consistía casi únicamente en escribir el código, bajo solo el conocimiento de los requisitos y con una modesta fase de análisis y diseño.

## Referencias históricas

---

El trabajo de Ada Lovelace, hija de Anabella Milbanke Byron y Lord Byron, que realizó para la máquina de Babbage le hizo ganarse el título de *primera programadora de computadoras* del mundo, aunque Babbage nunca completó la construcción de la máquina. El nombre del lenguaje de programación Ada fue escogido como homenaje a esta mujer programadora.

## Objetivos de la programación

---

La programación debe perseguir la obtención de programas de calidad. Para ello se establece una serie de factores que determinan la calidad de un programa. Algunos de los factores de calidad más importantes son los siguientes:

- Correctitud. Un programa es correcto si hace lo que debe hacer tal y como se estableció en las fases previas a su desarrollo. Para determinar si un programa hace lo que debe, es muy importante especificar claramente qué debe hacer el programa antes de su desarrollo y, una vez acabado, compararlo con lo que realmente hace. Al verificar este comportamiento está cumpliendo dicho objetivo.
- Claridad. Es muy importante que el programa sea lo más claro y legible posible, para facilitar tanto su desarrollo como su posterior mantenimiento. Al elaborar un programa se debe intentar que su estructura sea sencilla y coherente, así como cuidar el estilo de programación. De esta forma se ve facilitado el trabajo del programador, tanto en la fase de creación como en las fases posteriores de corrección de errores, ampliaciones, modificaciones, etc. Fases que pueden ser realizadas incluso por otro programador, con lo cual la claridad es aún más necesaria para que otros puedan continuar el trabajo fácilmente. Algunos programadores llegan incluso a utilizar Arte ASCII para delimitar secciones de código; una práctica común es realizar aclaraciones en el mismo código fuente utilizando *líneas de comentarios*. Contrariamente, algunos programadores realizan acciones que tienden a introducir confusión para impedir un análisis cómodo a otros programadores, recurren al uso de código ofuscado,
- Eficiencia. Se trata de que el programa, además de realizar aquello para lo que fue creado (es decir, que sea correcto), lo haga gestionando de la mejor forma posible los recursos que utiliza. Normalmente, al hablar de eficiencia de un programa, se suele hacer referencia al tiempo que tarda en realizar la tarea para la que ha sido creado y a la cantidad de memoria que necesita, pero hay otros recursos que también pueden ser de consideración para mejorar la eficiencia de un programa, dependiendo de su naturaleza (espacio en disco que utiliza, tráfico en la red que genera, etc.).
- Portabilidad. Un programa es portable cuando tiene la capacidad de poder ejecutarse en una plataforma, ya sea hardware o software, diferente a aquella en la que se desarrolló. La portabilidad es una característica muy deseable para un programa, ya que permite, por ejemplo, a un programa que se ha elaborado para el sistema GNU/Linux que también pueda ejecutarse en la familia de sistemas operativos Windows. Consecuentemente el programa puede llegar a más usuarios.

## Ciclo de vida del software

---


El término ciclo de vida del software describe el desarrollo de software, desde la fase inicial hasta la fase final, incluyendo su estado funcional. El propósito es definir las distintas fases intermedias que se requieren para validar el desarrollo de la aplicación, es decir, para garantizar que el software cumpla los requisitos para la aplicación y verificación de los procedimientos de desarrollo: se asegura que los métodos utilizados son apropiados. Estos métodos se originan en el hecho de que es muy costoso corregir los errores que se detectan tarde dentro de la fase de implementación (programación propiamente dicha), o peor aún, durante la fase funcional. En el modelo de ciclo de vida se intenta que los errores se detecten lo antes posible y por lo tanto, permite a los desarrolladores concentrarse en la calidad del software, en los plazos de implementación y en los costos asociados. El ciclo de vida básico de un software consta de, al menos, los siguientes procedimientos:

- Análisis de requisitos, viabilidad de diseño y especificación de funciones definidas en lenguaje de programación
- Análisis de la arquitectura en la creación, desarrollo, corrección e implementación del sistema.
- Pruebas en la integración de módulos, y subprograma(s) con cada conjunto o subconjunto.
- Pruebas beta o de validación que garanticen que en el procedimiento de ejecución del software se cumple con todas las especificaciones originales.
- Mantenimiento de corrección de errores y restricciones.
- Documentación de toda la información.

El orden y la presencia de cada uno de estos procedimientos dependen del tipo de modelo de ciclo de vida acordado entre el cliente y el equipo de desarrolladores. En el caso del software libre se tiene un ciclo de vida mucho más dinámico, puesto que muchos programadores trabajan en simultáneo desarrollando sus eliminaciones.

## Véase también

---

-  Portal:Programación. Contenido relacionado con **Programación**.
- Wikiproyecto:Informática/Programación
- error de software
- filosofías del desarrollo de software
- historia de la ingeniería del software
- ingeniería en computación
- ingeniería en informática
- Línea de código fuente
- lenguaje de programación
- programación automática
- programación dirigida por eventos
- programación estructurada
- programación extrema
- programación en pareja
- programación dinámica
- programación orientada a objetos
- pruebas de software
- software




## Referencias

---

1. Real Academia Española y Asociación de Academias de la Lengua Española (2014). «programar» (<http://dle.rae.es/programar>). *Diccionario de la lengua española* (23.ª edición). Madrid: Espasa. ISBN 978-84-670-4189-7.
2. JOSÉ LUIS LÓPEZ (Digital Marketing\_) (10 de diciembre de 2019). «Aprender por Internet» (<https://www.jluislopez.es/aprender-por-internet/>). Consultado el 10 de diciembre de 2019.
3. Laboda, Xavier; Josep Galimany, Rosa María Pena, Antoni Gual (1985). «Software». *Biblioteca práctica de la computación*. Barcelona: Ediciones Océano-Éxito, S.A.
4. «Lenguajes de programación» ([http://informatica.uv.es/iiguia/AED/oldwww/2004\\_05/AED.Tema.02.pdf](http://informatica.uv.es/iiguia/AED/oldwww/2004_05/AED.Tema.02.pdf)).

## Enlaces externos

---

-  [Wikimedia Commons](#) alberga una categoría multimedia sobre **Programación**.
-  [Wikcionario](#) tiene definiciones y otra información sobre **programación**.
-  [Wikilibros](#) alberga un libro o manual sobre **Fundamentos de programación**.

---

Obtenido de «<https://es.wikipedia.org/w/index.php?title=Programación&oldid=137034888>»

---

Esta página se editó por última vez el 15 jul 2021 a las 21:36.

El texto está disponible bajo la Licencia Creative Commons Atribución Compartir Igual 3.0; pueden aplicarse cláusulas adicionales. Al usar este sitio, usted acepta nuestros términos de uso y nuestra política de privacidad. Wikipedia® es una marca registrada de la Fundación Wikimedia, Inc., una organización sin ánimo de lucro.