El lenguaje entendido por una computadora se conoce como [código máquina](https://es.wikipedia.org/wiki/C%C3%B3digo_m%C3%A1quina). Consiste en secuencias de instrucciones básicas que el procesador reconoce, codificadas como cadenas de números 1 y 0 ([sistema binario](https://es.wikipedia.org/wiki/Sistema_binario)). En los primeros tiempos de la computación se programaba directamente en código máquina. Escribir programas así resultaba demasiado complicado, también era difícil entenderlos y mantenerlos una vez escritos. Con el tiempo, se fueron desarrollando herramientas para facilitar el trabajo.

Los primeros científicos que trabajaron en el área decidieron reemplazar las secuencias de unos y ceros por [mnemónicos](https://es.wikipedia.org/wiki/Mnem%C3%B3nico), que son abreviaturas en [inglés](https://es.wikipedia.org/wiki/Idioma_ingl%C3%A9s) de la función que cumple una instrucción de procesador. Por ejemplo, para sumar se podría usar la letra A de la palabra inglesa *add* (añadir). Crearon así una familia de lenguajes de mayor nivel, que se conocen como [lenguaje ensamblador](https://es.wikipedia.org/wiki/Lenguaje_ensamblador) o simplemente ensamblador (en inglés, *assembly*). Con el tiempo los ensambladores incorporaron facilidades adicionales, pero siempre manteniendo una correspondencia directa con las instrucciones de procesador. A nivel conceptual, entonces, programar en ensamblador es muy similar a hacerlo en lenguaje máquina, solo que de una forma más amigable.

A medida que la complejidad de las tareas que realizaban las computadoras aumentaba, el lenguaje ensamblador fue mostrando limitaciones. Para hacer un programa había que conocer en detalle el funcionamiento de la computadora donde se iba a ejecutar, qué instrucciones proveía y cómo emplearlas. A veces las instrucciones eran demasiado básicas, por ejemplo podía haber una para sumar dos números pero no para multiplicar, y entonces era necesario programar un algoritmo que realizara la multiplicación con base en instrucciones más básicas. Otras veces, la forma de emplear las instrucciones era engorrosa. Además, si se usaba otro modelo de computadora, en muchos casos había que reescribir el programa con otras instrucciones. El siguiente paso fue crear los [lenguajes de alto nivel](https://es.wikipedia.org/wiki/Lenguaje_de_alto_nivel).

Una vez que se termina de escribir un programa, es necesario de alguna forma traducirlo a lenguaje máquina, que es lo único que entiende el procesador. Esta tarea es automática, por medio de un programa adicional que toma el código escrito y lo procesa. Hay distintos enfoques para este procesamiento. El enfoque clásico se llama [compilación](https://es.wikipedia.org/wiki/Compilador): el programa toma el código en un lenguaje y genera código en el otro; al programa traductor se lo llama compilador. En general se habla de compilación y compiladores cuando el lenguaje de origen es de alto nivel; si la traducción es desde lenguaje ensamblador, se llama ensamblado y el programa se llama ensamblador (hay que distinguir el lenguaje ensamblador del programa ensamblador; en inglés es más claro, son *assembly language* y *assembler* respectivamente).[[3]](https://es.wikipedia.org/wiki/Programaci%C3%B3n#cite_note-3)​ Generalmente existe una fase posterior a la compilación denominada [enlace](https://es.wikipedia.org/wiki/Enlazador) o enlazado (*linking* en inglés). Los programas pueden escribirse en partes separadas y además pueden usar recursos provistos por [bibliotecas](https://es.wikipedia.org/wiki/Biblioteca_(inform%C3%A1tica)). El enlazado, realizado por un programa llamado enlazador, combina todos los componentes y así genera un programa ejecutable completo.

En algunos lenguajes de programación, puede usarse un enfoque diferente que no requiera compilación y enlace: un programa llamado [intérprete](https://es.wikipedia.org/wiki/Int%C3%A9rprete_(inform%C3%A1tica)) va leyendo el código y realizando en el momento las acciones que haría el programa. Se evita generar código separado y la experiencia es que se está ejecutando el código en el lenguaje de alto nivel, a pesar de que el procesador no lo entienda de forma nativa.

**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](https://es.wikipedia.org/wiki/Lengua_natural) acotada (en [inglés](https://es.wikipedia.org/wiki/Idioma_ingl%C3%A9s)); y que además tienen la particularidad de una reducida ambigüedad.

En los [lenguajes de programación](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) se distinguen diversos elementos entre los que se incluyen el léxico propio del lenguaje y las reglas semánticas y sintácticas. Dentro del léxico, generalmente se utilizan símbolos y palabras con funciones específicas dentro del lenguaje. Estas palabras suelen tomarse del inglés y no se las puede utilizar de manera diferente: son las denominadas [palabras reservadas](https://es.wikipedia.org/wiki/Palabra_reservada). Otra particularidad de los lenguajes es el permitir a los programadores el uso de comentarios: frases o párrafos sin funcionalidad en el programa, que los compiladores o intérpretes descartan y solo están destinados a ser leídos por personas; así se pueden dejar explicaciones que ayuden a entender el código a quien lo lea.[[4]](https://es.wikipedia.org/wiki/Programaci%C3%B3n#cite_note-4)​

**Programas y algoritmos**

Un [algoritmo](https://es.wikipedia.org/wiki/Algoritmo) es una secuencia no [ambigua](https://es.wikipedia.org/wiki/Ambig%C3%BCedad), finita y ordenada de instrucciones que han de seguirse para resolver un determinado problema.[[1]](https://es.wikipedia.org/wiki/Programaci%C3%B3n#cite_note-:0-1)​ 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](https://es.wikipedia.org/wiki/Diagrama_de_flujo), en forma de código como en [pseudocódigo](https://es.wikipedia.org/wiki/Pseudoc%C3%B3digo) 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](https://es.wikipedia.org/wiki/Niklaus_Wirth), un programa está formado por los [algoritmos](https://es.wikipedia.org/wiki/Algoritmo) y [estructuras de datos](https://es.wikipedia.org/wiki/Estructuras_de_datos).

La programación puede seguir muchos enfoques, o [paradigmas](https://es.wikipedia.org/wiki/Paradigma_de_programaci%C3%B3n), es decir, diversas [maneras](https://es.wikipedia.org/wiki/Linear_probing) de formular la resolución de un problema dado. Algunos de los principales paradigmas de programación son:

* [Programación declarativa](https://es.wikipedia.org/wiki/Programacion_declarativa)
* [Programación imperativa](https://es.wikipedia.org/wiki/Programaci%C3%B3n_imperativa)
* [Programación estructurada](https://es.wikipedia.org/wiki/Programaci%C3%B3n_estructurada)
* [Programación modular](https://es.wikipedia.org/wiki/Programaci%C3%B3n_modular)
* [Programación orientada a objetos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos)
* [Programación orientada a eventos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_dirigida_por_eventos)

**Compilación**

[](https://es.wikipedia.org/wiki/Archivo:Programmer_writing_code_with_Unit_Tests.jpg)Programadora escribiendo líneas de código en JAVA con JUnit.

El programa escrito en un [lenguaje de programación](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) de alto nivel (fácilmente comprensible por el [programador](https://es.wikipedia.org/wiki/Programador)) es llamado *programa fuente* y no se puede ejecutar directamente en una computadora. La opción más común es [compilar](https://es.wikipedia.org/wiki/Compilador) 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.](https://es.wikipedia.org/wiki/Int%C3%A9rprete_inform%C3%A1tico) Algunos lenguajes, tal como [BASIC](https://es.wikipedia.org/wiki/BASIC), disponen de ambas formas de ejecución, lo cual facilita la tarea de depuración y prueba del programa.

El código fuente del [programa](https://es.wikipedia.org/wiki/Programa_(computaci%C3%B3n)) se debe someter a un [proceso de traducción](https://es.wikipedia.org/wiki/Proceso_de_traducci%C3%B3n_de_programas) para convertirlo a lenguaje máquina o bien a un código intermedio, generando así un módulo denominado "objeto". A este proceso se le llama [*compilación*](https://es.wikipedia.org/wiki/Compilaci%C3%B3n).

Habitualmente la creación de un programa [ejecutable](https://es.wikipedia.org/wiki/Ejecutable) (un típico.exe para [Microsoft Windows](https://es.wikipedia.org/wiki/Microsoft_Windows) o [DOS](https://es.wikipedia.org/wiki/DOS)) conlleva dos pasos: el primer paso se llama compilación (propiamente dicho) y traduce el código fuente, escrito en un [lenguaje de programación](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) 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*](https://es.wikipedia.org/wiki/Enlazador) en el cual se enlaza el código de bajo nivel generado de todos los ficheros y subprogramas que se han mandado a 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](https://es.wikipedia.org/wiki/C%C3%B3digo_objeto) a [código máquina](https://es.wikipedia.org/wiki/C%C3%B3digo_m%C3%A1quina), 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](https://es.wikipedia.org/wiki/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](https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)), [C](https://es.wikipedia.org/wiki/C_(lenguaje_de_programaci%C3%B3n)), [C++](https://es.wikipedia.org/wiki/C%2B%2B) y [ensamblador](https://es.wikipedia.org/wiki/Lenguaje_ensamblador), que se podrían compilar de forma independiente y luego combinarse para formar un único módulo [ejecutable](https://es.wikipedia.org/wiki/Ejecutable).

**Programación e ingeniería del software**

Artículo principal: [*Ingeniería del software*](https://es.wikipedia.org/wiki/Ingenier%C3%ADa_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](https://es.wikipedia.org/wiki/Ingenier%C3%ADa), 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](https://es.wikipedia.org/wiki/Requisito_(sistemas)). 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.