

TIC01 INTRODUCCIÓN A LA PROGRAMACIÓN

- 1 Introducción.**
- 2 Formas de resolver problemas. Algoritmos y heurísticas.**
- 3 El software de programación. Los lenguajes de programación.**
- 4 Clasificación de los lenguajes de programación.**
 - 4.1 Lenguajes de bajo nivel.**
 - 4.1.1 Lenguaje máquina.**
 - 4.1.2 Lenguaje ensamblador.**
 - 4.2 Lenguajes de alto nivel.**
- 5 Compiladores e intérpretes**
- 6 Los entornos de desarrollo integrados (IDE).**

1.- INTRODUCCION

La información es todo aquello que permite adquirir cualquier tipo de conocimiento. Según la RAE, la información (RAE) es la comunicación o adquisición de conocimientos que permiten ampliar o precisar los que se poseen sobre una materia determinada.

Una información consta de una serie de datos. En ocasiones, estos datos, pueden ser utilizados directamente, y en otras, hay que realizar con ellos una serie de operaciones para obtener un resultado. Al conjunto de operaciones que se realizan sobre una cierta información con objeto de conseguir determinados resultados, se le denomina **tratamiento de la información**.

La **informática** (RAE) es el conjunto de conocimientos científicos y técnicas que hacen posible el tratamiento automático de la información por medio de ordenadores.

Concepto de ordenador

Se puede definir el **ordenador** como una máquina universal capaz de interpretar y ejecutar una serie de operaciones elementales (instrucciones), relativas al tratamiento de la información, y resolver cualquier tipo de aplicación o tarea gracias a la utilización automática de programas registrados en ella.

Según la RAE, un ordenador o computadora electrónica es una máquina electrónica que, mediante determinados programas, permite almacenar y tratar información, y resolver problemas de diversa índole.

Un ordenador se compone de un equipo físico o hardware y un equipo lógico o software. El equipo físico está formado por componentes electrónicos, que proporcionan el soporte necesario para la interpretación y ejecución de un pequeño repertorio de instrucciones elementales. El software (RAE) es el conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

El software es la razón por la que las personas utilizan ordenadores. Un ordenador sin el software es una máquina sin utilidad. Para que un ordenador pueda desarrollar determinadas tareas (mover el ratón, poner un texto en negrita, sumar dos números, etc.) hay que decirle lo que tiene que hacer. Al conjunto de instrucciones que dirigen a una computadora para que realice tareas específicas se conoce como **programa**.

Programa: Conjunto unitario de instrucciones que permite a una computadora realizar funciones diversas, como el tratamiento de textos, el diseño de gráficos, la resolución de problemas matemáticos, el manejo de bancos de datos, etc.

Un programa sería una secuencia de instrucciones comprensibles por el ordenador que permiten realizar una determinada tarea con la finalidad de resolver un problema concreto.

Aunque es capaz de llevar a cabo tareas muy complejas, un ordenador sólo puede realizar una serie de operaciones simples y obedecer instrucciones en lenguaje máquina. Una

instrucción en lenguaje máquina no es más que una cadena de ceros y unos que expresan, de forma codificada, las únicas órdenes que un ordenador es capaz de entender. La potencia de un ordenador no radica tanto en la complejidad de las instrucciones que es capaz de ejecutar, que más bien son muy sencillas, sino en la gran velocidad de ejecución de las mismas.

Como trabajar en lenguaje máquina es muy engorroso para los humanos, se desarrollaron los lenguajes de programación para simplificar el proceso de escritura de instrucciones que las computadoras pueden utilizar.

Los lenguajes de programación presentan características comunes a todos ellos. Para que el ordenador entienda lo que se le dice en un lenguaje de programación hacen falta unos programas (compiladores e intérpretes) que traduzcan las instrucciones escritas en un determinado lenguaje de programación a lenguaje máquina.

2. FORMAS DE RESOLVER PROBLEMAS. ALGORITMOS Y HEURÍSTICAS.

Un ordenador es una máquina capaz de interpretar y ejecutar una serie de operaciones relacionadas al tratamiento de la información.

Entre las operaciones que se puede realizar con la información, en algunas de ellas, por ejemplo las de tipo aritmético, sólo se aplican unas reglas determinadas, que siempre son las mismas, y que pueden ser calificadas como mecánicas. En otras, por el contrario, resulta difícil encontrar una secuencia de instrucciones, que transformen una determinada operación en mecánica.

Algoritmos:

Al conjunto de operaciones que permiten realizar de modo automático cualquier operación se le denomina **algoritmo**. Un algoritmo es una serie de instrucciones paso a paso que, cuando se siguen, producen un resultado conocido o esperado. Conociendo el algoritmo

y los datos, cualquier operación se puede transformar en mecánica. Sin embargo, no todos los procesos son susceptibles de mecanizarse, ya que no siempre se puede hallar su algoritmo.

Algoritmo: secuencia ordenada y finita de acciones, exenta de ambigüedades y que puede realizar un ordenador y que lleva a la solución de un problema dado.

Heurística:

Algunas veces no existe un algoritmo para resolver un cierto problema, o el algoritmo es tan complejo o lento que no puede codificarse o ejecutarse. En estos casos, los programadores utilizan la heurística para ayudar a realizar tareas o resolver problemas. La heurística es un conjunto de pasos para encontrar la solución de un problema. Pero, a diferencia de los algoritmos, una solución heurística no proporciona una garantía de encontrar la mejor solución posible. La heurística ofrece posibilidades de encontrar una solución, aunque no necesariamente la mejor.

La heurística no suele utilizarse en las aplicaciones comunes (hojas de cálculo, procesadores de texto, etc.) debido a que muy pocas veces se requiere que se realice una tarea para la cual no existe una solución correcta, pero puede ser habitual en los programas más avanzados que trabajan con cantidades enormes de datos en forma compleja (por ejemplo, software que ayuda a pronosticar el clima). Algunos programas (por ejemplo, los antivirus) utilizan tanto la heurística como los algoritmos.

3.- EL SOFTWARE DE PROGRAMACION. LENGUAJES DE PROGRAMACION

El software se puede clasificar de acuerdo a las siguientes categorías:

- **Software del sistema:** el software del sistema incluye cualquier programa que controle el hardware de la computadora o que se puede utilizar para dar mantenimiento a la computadora de alguna forma que la haga funcionar con mayor

eficiencia. Dentro del software del sistema, se pueden englobar los sistemas operativos, las herramientas, los drivers, etc.

- **Software de programación:** el software de programación incluye los programas que utilizan los programadores para crear nuevos programas. Un lenguaje de programación es un conjunto de palabras clave o instrucciones y unas reglas sintácticas que indican como construir los programas. Entre los lenguajes de programación más conocidos se pueden citar C, C++, Java, etc.
- **Software de aplicación:** el software de aplicación resuelve problemas específicos de los usuarios. El software de aplicación le dice a la computadora la forma en que debe llevar a cabo tareas específicas para el usuario, como por ejemplo, procesar textos o dibujar. Algunos ejemplos de software de aplicación serían los procesadores de textos, las hojas de cálculo, los programas de retoque fotográfico, etc.

El software de programación incluye los programas que utilizan los programadores para crear nuevos programas. Los programas se crean utilizando un lenguaje de programación.

Un **lenguaje de programación** es un conjunto de palabras clave o instrucciones y unas reglas sintácticas que indican como construir los programas.

Los lenguajes de programación

Un **programa** es una secuencia de instrucciones comprensibles por el ordenador que le permiten realizar una determinada tarea con la finalidad de resolver un problema concreto.

El objetivo de la programación (programar) consiste en crear un conjunto de instrucciones para el ordenador. El programador es quien crea ese conjunto o lista de instrucciones. Esta lista se conoce como código y el proceso de escritura de la lista se conoce

como codificación. La programación es un proceso que permite al programador crear algo nuevo y resolver problemas.

Código: El término código se refiere a las declaraciones escritas en cualquier lenguaje de programación, ya sea código máquina o un lenguaje de alto nivel.

Los programadores utilizan los lenguajes de programación para crear el código fuente de los programas. Posteriormente, convierten el código fuente en código máquina (o código objeto) para la computadora.

Aunque existen múltiples herramientas de desarrollo que difieren enormemente en cuanto a capacidad, flexibilidad y facilidad de uso, todos los lenguajes de programación comparten una serie de características comunes. Algunas de estas características son las siguientes:

- Existe un conjunto de reglas obligatorias que el programador debe seguir.
- Se utilizan símbolos especiales.
- No se permiten errores léxicos, sintácticos ni semánticos.
- Hay que proporcionar la información en un orden y estructura específicos.

Estas reglas constituyen la sintaxis del lenguaje de programación y pueden variar mucho de un lenguaje a otro. Cuando no se cumplen estas reglas, el compilador o intérprete emitirá errores y advertencias, no podrá traducir el código fuente y fallará en la creación del código objeto.

4.- CLASIFICACIÓN DE LOS LENGUAJES DE PROGRAMACION

Según la RAE, un lenguaje de programación es un conjunto de signos y reglas que permite la comunicación con una computadora.

Según su nivel de abstracción, los lenguajes de programación se clasifican en:

- **Lenguajes de alto nivel**, o lenguajes orientados al problema.
- **Lenguajes de bajo nivel**, o lenguajes orientados al ordenador.

4.1. Lenguajes de bajo nivel

Los lenguajes de bajo nivel se encuentran totalmente vinculados al hardware, es decir, a la estructura interna del ordenador. Están diseñados para poder sacar partido directamente a la máquina y no para hacer frente a la solución de ningún tipo de problema específico.

Bajo esta denominación se engloban tanto el lenguaje máquina como el lenguaje ensamblador. Ambos se analizan a continuación.

4.1.1. Lenguaje máquina

El lenguaje máquina fue el lenguaje que utilizó el hombre para la programación de los primeros ordenadores. Una instrucción en lenguaje máquina no es más que una cadena de ceros y unos que expresan, de forma codificada, las únicas órdenes que un ordenador es capaz de entender.

Los lenguajes máquina son los lenguajes más básicos. Los lenguajes máquina son muy dependientes del diseño del hardware (Pentium, Macintosh, etc.) y las instrucciones se crean en forma de código máquina (ceros y unos). Una computadora sólo entiende su lenguaje máquina nativo, los comandos de su conjunto de instrucciones. Estos comandos le indican a la computadora que realice operaciones elementales como cargar, almacenar, sumar y restar.

Cualquier programa, para ser ejecutado en un ordenador, debe ser traducido a lenguaje máquina. Si un programa ha sido creado en otro lenguaje de programación, deberá sufrir un proceso de traducción antes de ser ejecutado.

4.1.2. Lenguaje ensamblador

El lenguaje ensamblador surgió con la idea de evitar las dificultades que presenta trabajar en lenguaje máquina. En términos generales, la misión de un lenguaje ensamblador es la de simplificar la programación de un determinado ordenador, manteniendo un control directo sobre el hardware del mismo. Los lenguajes ensamblador, aunque siguen siendo altamente detallados y arduos, son mucho más sencillos de usar que el lenguaje máquina. Las simplificaciones aportadas por los lenguajes ensambladores provienen fundamentalmente de dos aspectos:

- El empleo de códigos nemotécnicos (abreviaciones cortas del idioma inglés) para representar las instrucciones.
- El empleo de nombres simbólicos para designar los datos y las referencias a memoria.

Otra característica que distingue a los lenguajes ensambladores de los lenguajes de alto nivel es que, el programador en ensamblador tiene acceso a todos los recursos e instrucciones de la máquina a la que se aplica, no así el programador en un lenguaje de alto nivel.

Para desarrollar software con un lenguaje ensamblador hay que utilizar un editor de texto para crear los archivos de código. Para convertir el código fuente en código objeto se utiliza un programa de traducción (traductor) especial llamado ensamblador.



Los programas sólo se escriben en lenguaje ensamblador cuando se necesita un código eficiente y rápido (por ejemplo, en juegos de acción), donde la velocidad del programa es muy importante.

4.2. Lenguajes de alto nivel

El código máquina (ceros y unos) es el único lenguaje que una computadora entiende. Sin embargo, las personas tienen dificultad para entender el código máquina. Para hacer que el desarrollo de software sea más sencillo, se han desarrollado lenguajes de programación cada vez más sofisticados. Esta evolución ha permitido que los programadores puedan pensar sobre sus programas en términos humanos en lugar de en términos de computadoras.

Un lenguaje de programación de alto nivel es un lenguaje orientado hacia la resolución de una determinada clase de problemas.

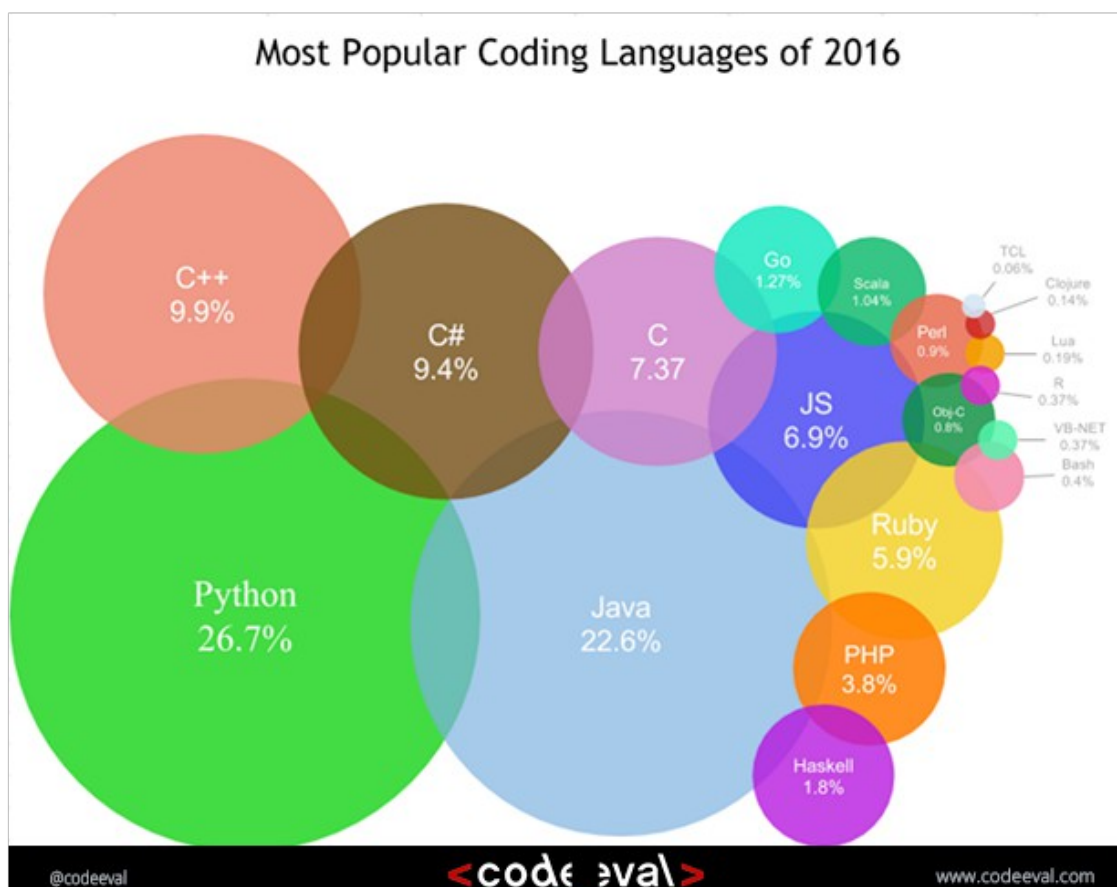
Los lenguajes de alto nivel fueron desarrollados para hacer que la programación fuera mucho más sencilla. Se les llama lenguajes de alto nivel porque su sintaxis es más parecida al idioma humano que el código de los lenguajes ensamblador o máquina. Los lenguajes de programación de alto nivel utilizan palabras familiares en lugar de las cadenas de dígitos que forman las instrucciones del lenguaje máquina. Las personas pueden leer, escribir y entender programas de manera mucho más sencilla cuando utilizan un lenguaje de nivel más alto.

Un lenguaje de alto nivel es independiente de la arquitectura del ordenador que lo soporta. Esto presenta dos ventajas principales:

- El programador no tiene que saber nada acerca del ordenador en el que se ejecutará el programa.
- Los programas son transportables, es decir, el mismo programa puede (en teoría) ejecutarse en diferentes tipos de ordenadores.

Otra característica de los lenguajes de alto nivel es que, generalmente, son más cortos que sus equivalentes en bajo nivel.

Entre los lenguajes de programación más conocidos se pueden citar: C, C++, Ada, Java, Pascal, Prolog, Lisp, Visual Basic, Python, etc.



Lenguajes de programación más utilizados (Ranking de Codeeval)

<http://blog.codeeval.com/>

5. COMPILADORES E INTÉRPRETES

Como ya hemos dicho, las computadoras utilizan el sistema binario de numeración, el cual utiliza ceros y unos. Cualquier comando de software que afecte directamente al hardware

debe estar escrito en el sistema binario de numeración. Como estos ceros y unos forman el lenguaje del hardware de las computadoras, este código se conoce como código máquina o lenguaje máquina. El lenguaje máquina está formado por ceros y unos.

Actualmente, los programadores no trabajan ni piensan en lenguaje máquina. Los programadores utilizan lenguajes de programación en lugar de lenguajes binarios. Los lenguajes de programación permiten al programador escribir un programa utilizando una variante del idioma inglés clásico. Los resultados se guardan en un archivo y se llaman código fuente.

Después de crear un fragmento de código fuente, hay que convertirlo en código máquina (una serie de ceros y unos) antes de que pueda ejecutarse en una computadora. El trabajo de convertir el código fuente en código máquina se puede hacer con dos tipos de programas:

- **Compiladores:** (C++, Pascal, etc.): un compilador convierte todo el código fuente en código máquina y crea un archivo ejecutable. El resultado del compilador se conoce como código objeto. Una vez compilado el programa, se convierte en un archivo ejecutable independiente que no necesita el compilador para ejecutarse.



- **Intérpretes:** (LISP, BASIC, Visual Basic, etc.): un intérprete también convierte el código fuente en código máquina. Sin embargo, en lugar de crear un archivo de código objeto ejecutable, lo traduce y luego ejecuta cada línea del programa de una en una. Al traducir el código sobre la marcha, los intérpretes tienen una flexibilidad que no tienen los compiladores. Por otro lado, el código interpretado se ejecuta de manera más lenta

que el código compilado.

Las instrucciones deben ser traducidas al lenguaje máquina para que la computadora pueda entenderlas. Una línea de un lenguaje de alto nivel se suele traducir en muchas líneas de lenguaje máquina. Esto hace que sea más rápido escribir en un lenguaje de alto nivel, pero sacrifica parte del control sobre el código que se produce.

6.- LOS ENTORNOS DE DESARROLLO INTEGRADOS (IDEs)

Actualmente, la práctica totalidad de los programas que se desarrollan, disponen de un interfaz de gráfico, que consiste en un conjunto de ventanas y menús que permiten al usuario interactuar con el software y proporcionarle una experiencia lo más satisfactoria posible.

Dadas las crecientes exigencias de los usuarios por disponer de interfaces de usuario amigables surgieron hace unos años distintos programas, conocidos como entornos integrados de desarrollo (IDEs) orientados a facilitar el trabajo al programador que debía desarrollar dicho software. La mayoría de los IDEs tienen una vista de diseño en la que el programador puede construir de manera gráfica las distintas ventanas y menús, y añadir los componentes gráficos (cajas de texto, botones, etc.) que aparecen en el programa y una vista de codificación en la que el programador accede al código fuente del programa.

La introducción de la posibilidad de disponer de la vista de diseño en los IDEs fue un gran avance para los programadores y les permitió desarrollar las aplicaciones de forma más rápida, con menos esfuerzo, y ofrecer programas de mayor calidad.

Algunos ejemplos de IDEs son: Netbeans, Eclipse, Dev-C++, BlueJ, JCreator, JBuilder, MS Visual Studio, JDeveloper, etc.