Lenguajes de programación
Cuando se programa, se indica una serie de instrucciones para que la computadora las ejecute. Para asegurarse que la computadora entienda las instrucciones, se han establecido lenguajes bien definidos para especificar y poder generar la comunicación con al computadora. Estos lenguajes tienen características similares a los lenguajes comunes que utilizan las personas para comunicarse unas con otras, pues cuentan con reglas y estructuras que deben seguirse.



A continuación se expondrá lo que son los lenguajes de computación, el papel que juegan dentro de la programación, herramientas que permiten traducir los algoritmos a instrucciones que puedan ser interpretadas y ejecutadas por las computadoras.
¿Qué es un Lenguaje de programación?
En computación, un lenguaje de programación es cualquier lenguaje artificial, el cual, se utiliza para definir adecuadamente una secuencia de instrucciones que puedan ser interpretadas y ejecutadas en una computadora. Se asume que las instrucciones así escritas son traducidas luego a un código que la máquina pueda “comprender”. El proceso de traducción es realizado normalmente por la computadora, usando un programa especializado para tal fin.

Los lenguajes de programación intentan conservar una similitud con el lenguaje humano, con la finalidad de que sean más naturales a quienes los usan. Establecen un conjunto de reglas sintácticas y semánticas, las cuales rigen la estructura del programa de computación que se escribe o edita. De esta forma, permiten a los programadores o desarrolladores, poder especificar de forma precisa los datos sobre los que se va a actuar, su almacenamiento, transmisión y demás acciones a realizar bajo las distintas circunstancias consideradas.

Uso de los lenguajes de programación:
Una vez que se identifica una tarea y se conoce el algoritmo para resolverla, el programador debe codificarlo en una lista de instrucciones, propias de algún lenguaje de computación. Los programas pueden ser escritos en cualquier lenguaje de la amplia gama disponibles. La escogencia del más adecuado, dependerá de factores como la experiencia del programador con un lenguaje, determinar cuál producirá el programa menos complicado, la flexibilidad y la compatibilidad del programa como resultado del lenguaje usado.

A continuación se presenta como ejemplo un sencillo programa fuente, escrito en C, el cual, es un lenguaje de programación que llegó a ser muy popular, y justo al lado se explica la acción que realizará la computadora en respuesta a los comandos del programa. El programa tiene como finalidad el cálculo del promedio de una lista de cinco números.

|  |  |
| --- | --- |
| Instrucciones en Lenguaje C | Acción de la computadora |
|    |    |
| int datos[ ] = { 25, 88, 20, 75, 62 }; | 1.- Define una matriz (datos) de números enteros y le asigna los valores 25, 88, 20, 75, 62. |
| int nDatos = 5; | 2.- Define la variable (nDatos) de tipo entero y le asigna el valor 5.  |
| int promedio () | 3.- Define la función (promedio) que devuelve un número entero. |
| { |  |
|    int suma; | 4.- Define la variable (suma) de tipo entero.  |
|    int prom; | 5.- Define la variable (prom) de tipo entero.  |
|    int cont; | 6.- Define la variable (cont) de tipo entero.  |
|    suma = 0; | 7.- Asigna a suma el valor 0.  |
|    for (cont=0; cont<nDatos; cont++) | 8.- Para cada elemento de la matriz datos, añade su valor a la variable suma. |
|       {  |  |
|       suma += datos [cont]; |  |
|       } |  |
|    prom=suma/nDatos; | 9.- Asigna a prom el resultado de dividir la variable suma por la variable nDatos. |
|    return (prom); | 10.- Devuelve como resultado de la función el contenido de la variable prom. |
| } |  |

Clasificación de los lenguajes de programación: Niveles y Generaciones:
En computación se suele hablar de niveles y generaciones de los lenguajes de programación. Respecto al nivel, se clasifican en lenguajes de bajo y alto nivel. Los lenguajes de bajo nivel, son aquellos que se asemejan al lenguaje empleado por la computadora, mientras que los lenguajes de alto nivel, son aquellos que se asemejan al lenguaje humano.


Niveles de los lenguajes de programación

Los lenguajes de alto nivel son resultado de una evolución, desde los lenguajes de bajo nivel. La computadora sólo es capaz de comprender su lenguaje máquina. Pero para las personas no es fácil entender y manejar este código máquina de combinaciones binarias. Las investigaciones dieron un primer paso que llevó a la evolución desde las hileras de números binarios del código máquina, a un lenguaje ensamblador, que las representaba como una serie de comandos más entendibles. Los lenguajes de nivel superior están orientados más hacia lo que se quiere realizar, que en la manera como se ejecutarán los comandos.
Niveles de los lenguajes de programación:
1.- Lenguajes de bajo nivel:
A nivel de circuitos, los microprocesadores sólo son capaces de procesar señales electrónicas binarias. Estas instrucciones binarias que se envían al microprocesador, se componen de series de unos y ceros, espaciadas en el tiempo de una forma determinada. Esta secuencia de señales se denomina código máquina. El código representa normalmente datos y números, así como instrucciones para manipularlos. Para hacer más fácil el manejo del código máquina, se dio a cada instrucción un mnemónico, como por ejemplo STORE (Almacenar), ADD (Adicionar) o JUMP (Saltar). Tal abstracción da como resultado el ensamblador, que es un lenguaje de muy bajo nivel basado en mnemónicos y que es específico para cada microprocesador.

De esta forma, son llamados lenguajes de bajo nivel, a aquellos constituidos por instrucciones en lenguaje máquina o próximos a él. Tales lenguajes permiten crear programas muy rápidos y eficientes, pero en contrapartida, estos lenguajes suelen ser difíciles de aprender, programar y depurar. Un hecho importante es, que los programas escritos en un bajo nivel, son prácticamente específicos para cada procesador, lo que implica, que si se quiere ejecutar el programa en otra máquina de arquitectura diferente, es necesario volver a escribir el programa desde el principio.

2.- Lenguajes de alto nivel:
Al pensar en una computadora, normalmente se concibe como una máquina que realiza tareas de procesamiento de texto y cálculos. Esta es una percepción bastante sintética y esquemática de entender la computadora. Existe un alto nivel de abstracción entre lo que se pide a la computadora y lo que realmente comprende. De un modo paralelo se presenta una relación compleja entre los lenguajes de alto nivel y el código máquina. Un programa escrito en un lenguaje de alto nivel, debe ser compilado o interpretado para traducir su código, en otro de bajo nivel (lenguaje máquina). Como ejemplo de lenguajes de alto nivel se tienen Pascal, BASIC, FORTRAN, COBOL, Java, C y C++.

En los lenguajes de alto nivel las instrucciones son independientes de la máquina, son más fáciles de aprender porque están formados por elementos de lenguajes naturales (del inglés). En BASIC, comandos como "IF CONTADOR = 10 THEN STOP" son usados para ordenar a la computadora, que pare si CONTADOR es igual a 10. Pero aunque las computadoras parecieran comprender el lenguaje natural, lo hacen de una manera bastante rígida y sistemática, lo cual podría parecer un poco frustrante e inspira la búsqueda de lenguajes más cercanos aún al natural.

Compiladores e Intérpretes:
Se ha señalado que la computadora sólo puede ejecutar instrucciones en código de máquina, por lo cual, todos los programas escritos en cualquiera de los lenguajes de programación deben ser convertidos o traducidos. Este proceso es llevado a cabo por unos programas especiales, los “compiladores” y los “interpretes”. Para darse una idea de cómo funcionan, es útil comparar su actuación con la de un profesional humano encargado de traducir. Un compilador equivale a que el traductor humano, partiendo de un texto, prepare otro texto nuevo e independiente, traducido a la otra lengua. En contraste, un intérprete equivale a que tal profesional traduzca de forma hablada, cada grupo de palabras o instrucción, a medida que la oye, sin dejarlas por escrito.
La traducción del código fuente, escrito en lenguaje ensamblador, a código máquina (o código objeto), no representa un proceso complicado y se lleva a cabo mediante un compilador. De manera similar, la traducción de código fuente de lenguajes de alto nivel puede ser realizada por un compilador, o también mediante un intérprete. Con base en el código fuente suministrado, el compilador crea una lista de instrucciones de código máquina:


Proceso de compilación

El código objeto. Este código objeto obtenido, es un programa rápido y ya listo para funcionar, pero está expuesto a poseer errores que pueden provocar fallas en la computadora. En el proceso de compilación pueden resultar varios archivos de código objeto, los cuales deben enlazarse para obtener un archivo ejecutable. Esto se logra mediante un programa llamado enlazador.
Por otra parte, los intérpretes recorren línea a línea el código fuente en el momento de su ejecución. Esto los hace más lentos, debido a la necesidad de traducir el programa mientras se ejecuta, pero a cambio, son más flexibles como entornos de programación y depuración. Es natural que los lenguajes de programación utilicen sólo uno de estos métodos para producir el código fuente, sin embargo muchos también disponen de versiones interpretadas y compiladas. En cuanto a las generaciones de lenguaje, se tienen cinco diferentes, considerando que cada generación sucesiva se establece con lenguajes más poderosos y fáciles de usar que los de la generación precedente. Los lenguajes calificados como de alto nivel, comienzan a partir de la tercera generación.



Generaciones de los lenguajes de programación


El lenguaje máquina consiste de códigos binarios

• Primera generación: Lenguaje máquina Son los lenguajes más básicos. Los lenguajes de máquina se representan en bytes, los cuales, están compuestos de bits, que son números binarios o que consisten únicamente de ceros (0) o unos (1). Estos valores pueden corresponder a que la electricidad esté encendida o apagada en la máquina, o a la presencia o ausencia de carga magnética en un medio de almacenamiento. A partir de estos dos estados se forman los esquemas de codificación (como el código ASCII) que permiten generar letras, números, signos de puntuación y caracteres especiales. Una computadora sólo es capaz de comprender su lenguaje máquina original, el conjunto de instrucciones para realizar sus operaciones elementales.
     • Segunda generación: lenguajes ensambladores:
       Están constituidos por nemotécnicos similares a palabras en idioma inglés. En primer lugar, se        crea un archivo fuente con las instrucciones que se desea que la computadora ejecute, usando        un editor, que es una especie de procesador de palabras. El código fuente es traducido al              lenguaje máquina mediante programas traductores (compiladores). Los programadores rara vez        escriben programas de tamaño relevante en este tipo de lenguajes, debido a que, a pesar que es        mucho más fácil que utilizar código de máquina, siguen siendo altamente detallados                      (instrucciones muy básicas). Sólo se hace en los casos donde la velocidad es clave (como en la        programación de juegos de video) y para afinar partes importantes de los programas que son        escritos en lenguajes superiores.

     • Tercera generación: Lenguajes de alto nivel:
       Son llamados de alto nivel porque se asemejan más al lenguaje que utilizan los humanos al        comunicarse. Fueron desarrollados con la finalidad de facilitar el proceso de programación.        Cuentan con comandos que se acercan más a las palabras de uso común en lugar de tener que        usar combinaciones binarias del código máquina o los comandos básicos del ensamblador. Estos        lenguajes hacen más fácil la lectura, escritura y comprensión de los programas, aunque de        manera semejante a los lenguajes ensambladores, deben ser convertidos a lenguaje máquina,        para poder ser usados en la computadora.
Los programas de la tercera generación son capaces de soportar programación estructurada, es decir, permiten el uso de estructuras de programación específicas para ramificaciones y ciclos en el flujo del programa. Por otro lado, resultan más claros puesto que usan palabras o comandos semejantes al lenguaje normal, lo que permite compartir el desarrollo entre diferentes programadores. Los programadores que conforman un equipo pueden leer el código realizado por los demás y comprender la lógica y el flujo de control del programa.


Los lenguajes de tercera generación crean
programas más claros y estructurados

Entre los lenguajes de tercera generación se incluyen:

- FORTRAN (Traductor de Fórmulas: Formula TRANslator), diseñado específicamente para     programas matemáticos y de ingeniería, motivo por el cual, no fue de amplio uso en computadoras     personales. Su uso se dirige especialmente a equipos de cómputo dedicados a la investigación y la     educación. La versión más actual (1990) fue denominada FORTRAN-90.

- COBOL (Lenguaje simbólico de programación, orientado hacia aplicaciones comerciales y de     gestión: Common Business Oriented Languaje) desarrollado en 1960, con el fin de solucionar las     incompatibilidades de los fabricantes de computadoras. Ganó amplia aceptación como lenguaje     estandarizado llegándose a utilizar ampliamente en los negocios, pero perdió seguidores en la     pasada década. Motivado al problema del año 2000 fueron requeridos programadores     especializados en este lenguaje para reprogramar las millones de líneas de código escritas en     COBOL para que tales programas continuasen funcionando.

- BASIC (Código de instrucción simbólica universal para principiantes: Begginners Allpurpose     Symbolic Instruction Code) comienza como una herramienta para enseñar sobre programación a     estudiantes. Fue desarrollado por Jhon Kemeny y Thomas Kurtz en 1964. Por su simplicidad se     volvió ampliamente popular y usado en educación y entre los programadores aficionados, pero no     ha llegado a ser un lenguaje viable para aplicaciones comerciales. Esto último se debe a que no     posee un amplio repertorio de herramientas como el que ofrecen otros lenguajes. Además, sus     compiladores no producen archivos ejecutables tan compactos, rápidos y eficientes como los que     producen otros lenguajes.

- PASCAL, nombrado en honor del inventor francés Blas Pascal, es introducido por Niklaus Wirth en     1971. El Pascal, buscaba superar las limitaciones de los otros lenguajes de programación y     demostrar la manera adecuada de implementar un lenguaje de cómputo. Ha llegado a considerarse     un excelente lenguaje de enseñanza. Las personas que se inician en la programación le consideran     fácil para implementar algoritmos, su compilador es estricto con el uso de programación     estructurada, asegurándose de que los errores sean señalados al principio.

- C, desarrollado a principios de los sesenta por Brian Kernighan y Dennis Ritchie, quienes buscaban     un mejor lenguaje de integrar con el sistema operativo UNIX, para que los usuarios pudiesen hacer     modificaciones y mejoras con facilidad. El código ejecutable obtenido del lenguaje C, es rápido y     eficiente. El C es poderoso, puesto que brinda la libertad de conseguir que se realice casi     cualquier cosa en una computadora. Esta es una de las razones por la cual se ha vuelto tan popular     entre quienes desarrollan software para aplicaciones comerciales profesionalmente. Su dificultad     estriba, en que no es tan sencillo de aprender en comparación con otros lenguajes.

- C++, Es un lenguaje de principios de los ochenta desarrollado por Bjarne Stroustrup. Es     extremadamente poderoso y eficiente igual que el C. Su aprendizaje implica aprender todo acerca     de C y además aprender sobre la programación orientada a objetos y su aplicación mediante el     C++. Desde su origen ha ido ganando terreno entre quienes desarrollaban en C.

- Java (1991) es desarrollado por Sun Microsistema. Cuando Internet comenzó a ganar popularidad     como red de comunicaciones a mediados de los noventa, Sun redirige a Java para convertirlo en un     ambiente de programación que permitiese crear sistemas interactivos y dinámicos (denominados     applets) para las páginas Web. La complejidad de Java es similar a C++, pero ha ido ganando     popularidad por la demanda de aplicaciones para páginas Web, incluso se estima que puede llegar     a desplazar a C++ como ambiente de programación.

     • Cuarta generación: Lenguajes de propósito especial:
       Estos lenguajes de cuarta generación (4GL) son usados en programación, de propósitos                      específicos, caracterizados por una mayor facilidad de uso comparados con los de la tercera        generación, permitiendo la creación de prototipos de una aplicación rápidamente. Los prototipos        permiten tener una idea del aspecto y funcionamiento de la aplicación antes que el código sea        terminado. Esto implica que quienes estén involucrados en el desarrollo de la aplicación pueden        aportar retroalimentación en aspectos como, estructura y diseño desde el principio del proceso.        A cambio de esta capacidad para trabajar más velozmente, debe sacrificarse parte de la               flexibilidad con la que se disponía con los lenguajes anteriores.

Muchos de estos lenguajes tienen capacidad para bases de datos, permitiendo crear programas que sirvan de enlace con las mismas. Los programas incluyen formas y cuadros para introducir datos, así como solicitar reportes de información de las bases de datos. En general, ahorran tiempo puesto que el código requerido para “conectar” los cuadros de diálogo y las formas se genera de forma automática.

Entre los programas de cuarta generación se tienen:

- Los programas de ambientes visuales: Existen versiones de programas populares como el Visual BASIC o el Visual C, los cuales, soportan características y métodos orientados a objetos y que permiten crear programas en un ambiente visual. Por ejemplo, si se desea ubicar un cuadro de texto en un formulario para entrada de datos sólo se requiere arrastrarlo desde la barra de herramientas hasta la posición deseada sobre el formulario. Con los programas de tercera generación se requiere escribir el código, indicando el tamaño y ubicación exacta del cuadro de texto.


La creación de formularios es más fácil en ambientes visuales.

En el ambiente visual, simplemente se ubica el cuadro en el lugar que se requiere y se da el tamaño adecuado arrastrando sus bordes con el ratón, el código se genera automáticamente. Esto facilita el desarrollo de la interfaz de usuario, acelerando y facilitando la creación de programas.
- Ambientes de autoría: Conformados por herramientas de programación para propósitos     especiales, se usan para crear multimedia, capacitación basada en computadora, páginas Web y     otros. Un ambiente de autoría es “Macromedia Director” el cual se usa para crear títulos     multimedia que combinen, cortos de música, texto, animación, gráficos y demás. Son ambientes de     desarrollo visuales en el cual gran parte del código se escribe automáticamente, pero la mayoría     de ambientes de autoría robustos también incluyen sus propios lenguajes, lenguajes de escritura,     que proporcionan un control extra sobre el producto final.

     • Quinta generación: Lenguajes naturales:
       Esta generación incluye la inteligencia artificial y sistemas expertos. En lugar de sólo ejecutar        un conjunto de órdenes, el objetivo de estos sistemas es “pensar” y anticipar las necesidades de        los usuarios. Estos sistemas se encuentran aún en desarrollo. Aunque los sistemas de inteligencia        artificial han demostrado ser más complicados para desarrollar de lo que se anticipaba, los               investigadores afirman que pronto serán capaces de tomar como entrada hechos y luego usar un        procesamiento de datos que formule una respuesta adecuada, de modo similar a como responden        los humanos.
Tabla 1.- Las cinco generaciones de lenguajes de programación

|  |  |  |
| --- | --- | --- |
| Generación | Nombre | Particularidad |
| Primera | De máquina | Específico para cada microprocesador, uso de código binario |
| Segunda | Ensamblador | Uso de nemotécnicos que abstraen del lenguaje máquina. |
| Tercera | De procedimientos | Lenguajes estructurados con comandos cercanos al lenguaje común. |
| Cuarta | Orientados a procesos | Programas orientados a problemas específicos. |
| Quinta | Natural | Incluye inteligencia artificial y sistemas expertos. |

Nuevas tendencias:
La disponibilidad de programas de calidad desarrollados más rápidamente, se ve acrecentada con el pasar del tiempo motivada por dos aspectos. Por un lado, la acumulación del conocimiento de cómo hacer las cosas, organizado mediante técnicas y metodologías bien definidas y estructuradas, que plantean alternativas como guías a seguir. Por otro lado, la disponibilidad de lenguajes de programación los cuales proveen mejores medios para que los desarrolladores sean capaces de aplicar su creatividad, ingenio y experiencia.
Una tendencia marcada desde hace algunos años, es la dependencia creciente de componentes de software reusable por parte de programadores y equipos de desarrollo. El enfoque de la programación orientada a objetos es muestra de ello, así como su incorporación en los lenguajes de programación. De tal modo, es muy probable que incluso lleguen a existir proveedores de objetos componentes de software, que los ofrezcan a los programadores, como en la actualidad se ofrecen componentes hardware. Entonces el enfoque del trabajo de los desarrolladores de software cambiará, tal vez, dividiéndolos en dos grupos.


En la programación cada vez, es más común
el uso de componentes reusables

Por un lado, quienes se encarguen de la integración de los objetos cuando se trata de desarrollar programas específicos, más que de desarrollar a detalle cada componente desde el principio. En tanto que los surtidores de tales componentes se especializan en desarrollar los objetos que les son demandados, que podrán ser componentes estándar o a la medida (con especificaciones únicas).
Otra gran influencia en el desarrollo de programas, es resultado de la expansión en el uso de Internet en lo personal y en lo comercial. Es posible que pronto se generalice el uso de programas los cuales se ejecuten en Internet y los desarrolladores produzcan programas que además de enviar y recibir mensajes y archivos, hagan uso de los inmensos recursos mundiales disponibles. Los programadores podrían estar desarrollando programas que se ejecuten en Internet, requiriendo entonces, lenguajes de programación más adaptados a este tipo de empresas.