INDICE

¿Qué es programación?

Se conoce como programación en ciencias de la computación a los pasos que se abordan para crear el código fuente de un programa informático. De acuerdo con estos pasos, el código se escribe, se prueba y se perfecciona.

El software informático es aquel ejecutado por el hardware o dispositivos de una computadora, permitiendo que el usuario acceda a funciones y operaciones de todo tipo. Así, la programación es una de las actividades más determinantes en el desarrollo de sistemas eficientes, rápidos y amigables para todo tipo de usuarios.

Se conoce como programadores a aquello encargados de desarrollar este código con instrucciones para que el software se comporte de una u otra manera de acuerdo con las órdenes que reciba. El lenguaje de programación es, por otro lado, la serie de parámetros y códigos de los que el programador se vale para desarrollar software. Existen distintos tipos de lenguajes, como el C, BASIC o Ruby.

Además, existe la ingeniería el software, que se dedica a desarrollar modelos de software para programas de gran envergadura.

Típicamente, para programar un software o aplicación, el programador debe en principio reconocer el principal problema o tarea a la que se destinará el programa, definir los requisitos y tipo de funcionamiento, diseñar la arquitectura, implementar el programa, implantarlo o instalarlo y, luego, perfeccionarlo sobre la base de pruebas y errores.

Hoy en día existen todo tipo de lenguajes de programación, algunos más sencillos o que tienen el propósito de facilitar la tarea de desarrollar pequeñas aplicaciones. Entre ellos, Ruby es uno de los más popularizados en los últimos años, desarrollado por un programador japonés y que combina sintaxis de distintos lenguajes como Python o Perl.

Así, prácticamente cualquier usuario informático puede adquirir algunas nociones de programación y desarrollar aplicaciones a su medida.

¿Qué es un lenguaje de programación?

Un lenguaje de programación es un idioma artificial diseñado para expresar computaciones que pueden ser llevadas a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana.[1] Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Al proceso por el cual se escribe, se prueba, se depura, se compila y se mantiene el código fuente de un programa informático se le llama programación.

También la palabra programación se define como el proceso de creación de un programa de computadora, mediante la aplicación de procedimientos lógicos, a través de los siguientes pasos:

- *El desarrollo lógico del programa para resolver un problema en particular.*
- *Escritura de la lógica del programa empleando un lenguaje de programación específico (codificación del programa).*
- *Ensamblaje o compilación del programa hasta convertirlo en lenguaje de máquina.*
- *Prueba y depuración del programa.*
- *Desarrollo de la documentación*

¿Qué es lógica computacional?

Muchas veces afirmamos que una cosa es lógica o un acontecimiento es lógico, y a lo que nos referimos en sí es a que, si a "nuestro entender" es razonable entonces en consecuencia será lógico. Sin embargo esta sólo es una definición intuitiva de lo que es, y además dependería de la perspectiva de cada quien.

La lógica es una ciencia que estudia los procesos del pensamiento, su estructura sus formas y relaciones. Teniendo como resultados respuestas claras, precisas y comprobables. Desechando todo resultado que no sean evidentes y precisos. Cuando hablamos de lógica, se nos viene a la mente todos los procedimientos o caminos más cortos para desarrollar o resolver un problema, teniendo la oportunidad de desechar ideas o razones erróneas dando oportunidad a otros pensamientos más factibles, hasta llegar al resultado válido. Con esto podemos tener el concepto siguiente : " La lógica es la ciencia de los principios de la inferencia formalmente válida".

La lógica estudia la metodología del pensar, todo lo que implica los procesos de la adquisición del conocimiento, y su estudio nos ayuda a formular métodos científicos, por lo que es instrumento en la investigación científica. Es importante notar que a pesar de toda la metodología que implica la lógica, no es necesario estudiarla para emplearla; esto es claro si tomamos en cuenta que todos los seres humanos realizamos de manera implícita operaciones lógicas en nuestras actividades cotidianas, sin que necesariamente nos demos cuenta de ello. Sin embargo el estudiarla nos sirve de herramienta para actuar con mayor eficacia en la práctica de nuestra profesión.

¿Clasificación de los lenguajes de programación?

LENGUAJE MÁQUINA.

Los ordenadores sólo entienden un lenguaje específico para cada máquina, que se denomina CÓDIGO MÁQUINA o Lenguaje Máquina. Este lenguaje utiliza un código binario (símbolos "0" y "1"). Las órdenes que se dan a un ordenador han de ir codificadas en instrucciones, y estas forman los programas. Las instrucciones tienen dos partes diferenciadas: código de operación y código(s) de operando(s):

CODOP CODOPERANDO(S)

En la primera, se codifica la operación que realiza la instrucción. Este código de operación siempre es único para cada instrucción. En la segunda se indica(n) la(s) dirección(es) de memoria en la que se encuentra el operando, hasta un máximo de tres, sobre el/(los) que se aplicará la operación.

Puesto que cada tipo de ordenador tiene su código máquina específico, para programar en este lenguaje el programador debe conocer la arquitectura física de la computadora con cierto detalle (registros de la CPU, palabras de memora,...). La estructura del lenguaje máquina está totalmente adaptada a los circuitos de la computadora y muy alejada del lenguaje que empleamos normalmente para expresar y analizar los problemas que hoy día son resolubles con la computadora. Por ejemplo, para hacer cálculos aritméticos disponemos de un "lenguaje" matemático fácil de comprender y claro, que no se parece en nada al código máquina necesario para hacer dichos cálculos.

Las ventajas de los lenguajes máquina son:

- Un programa escrito en lenguaje máquina es directamente interpretable por el procesador central. Una vez introducido el programa en la memoria principal de la computadora, no se necesitan transformaciones previas para ser ejecutado (como más adelante veremos que sí ocurre con los programas escritos en lenguajes de alto nivel).

- Los programas escritos en lenguaje máquina se ejecutan muy eficientemente (con rapidez), debido a que el usuario lo redacta específicamente para los circuitos que lo han de interpretar y ejecutar, y a que desde el código máquina se puede utilizar la totalidad de los recursos de la máquina.

Por contra, los lenguajes máquina tienen las siguientes características-inconvenientes:

- Las instrucciones son cadenas de ceros y unos, aunque estas cadenas se pueden introducir en la computadora mediante un código intermedio (octal o hexadecimal).

- Los datos se utilizan por medio de las direcciones de memoria donde se encuentran. En las instrucciones no aparecen nombres de variables (i, j aux...), sino que el programador debe asignar las direcciones de memoria para las variables y constantes del programa. Para realizar esta asignación se debe tener en cuenta la zona de memoria que ocupa el programa, para que no se solape con la zona en la que se almacenan las variables.

- El repertorio de instrucciones suele ser muy reducido y las instrucciones realizan operaciones muy simples.

En lenguaje máquina, hay varios tipos de instrucciones:

- De transferencia de información.

- De tratamiento o aritmético-lógicas y de desplazamiento

- De transferencias de control: bifurcaciones, saltos, llamadas a procedimientos y retornos de procedimientos.

Muchas computadoras, por ejemplo, no disponen de instrucciones específicas de multiplicar y dividir; en su lugar, el programador debe ingeniárselas para descomponer cada una de las operaciones que desee realizar en términos de las instrucciones elementales del repertorio máquina. Por ejemplo, para multiplicar, se deberá codificar un algoritmo que realice la multiplicación por medio de sumas, comparaciones, desplazamientos, etcétera.

- Existe muy poca elasticidad y versatilidad para la redacción de instrucciones. Estas tienen un formato rígido en cuanto a posición de los distintos campos que configuran la instrucción (código de operación, dirección o direcciones de memoria, códigos de puertos, etc.). El código de operación debe seleccionarse estrictamente entre los que figuran en una tabla o repertorio fijo. Además, un programa máquina no permite el uso de sentencias declarativas, existiendo sólo las instrucciones.

- El lenguaje máquina depende y está ligado íntimamente a la CPU del computador. Si dos computadoras tienen CPU's diferentes, tendrán distintos lenguajes máquina. En particular, dos

microcomputadoras con el mismo microprocesador e iguales circuitos de control, tienen igual lenguaje máquina. La dependencia del lenguaje máquina de la configuración de la CPU hace que los programas redactados en este lenguaje de programación sean poco transferibles o transportables de una computadora a otra.
- En un programa en código máquina, no pueden incluirse comentarios que faciliten la legibilidad del mismo. Además, debido a su representación totalmente numérica, es muy difícil de reconocer o interpretar por el usuario.

1.4. CLASIFICACIÓN DE LOS LENGUAJES DE PROGRAMACIÓN.

Tradicionalmente los lenguajes de programación se pueden clasificar atendiendo a varios factores:
- Según su "nivel". Hace referencia a lo próxima al hombre que esté la forma de expresar las sentencias:
** Lenguajes de bajo nivel y ensambladores (cercanos a la máquina).*
** Lenguajes de alto nivel (cercanos al hombre).*
- Según la relación traducción-ejecución.
** Compiladores.*
** Intérpretes.*
- Según su campo de aplicación:
** Aplicaciones Científicas. Predominan los algoritmos de cálculo numérico y matrices.*
** Aplicaciones de Procesamiento de Datos. Sobresalen las tareas relativas a la creación, mantenimiento, consulta y listado de datos. Estos datos se organizan en registros, ficheros y bases de datos.*
** Aplicaciones de Tratamiento de Textos. Llevan a cabo la manipulación de textos en lenguaje natural.*
** Aplicaciones en Inteligencia Artificial. Están constituidas por programas que emulan un comportamiento inteligente. Ej. juegos inteligentes (ajedrez, tres en raya, ...), robótica, sistemas expertos, ...*
** Aplicaciones de Programación de Sistemas. Como por ejemplo aquéllos que se utilizan para desarrollar los módulos de un Sistema Operativo, traductores de lenguajes, ...*
- Según el estilo de programación:
** Imperativos.*
** Declarativos.*
Nos centramos en esta última clasificación.

1.4.1. Lenguajes imperativos.

Los lenguajes imperativos o procedurales se basan en la asignación de valores. Se fundamentan en la utilización de variables para almacenar valores y en la realización de operaciones con los datos almacenados. La mayoría de los lenguajes son de este tipo (FORTRAN, BASIC, COBOL, PASCAL, C, etc.).
Son los que ya hemos analizado:
a) Lenguajes de alto nivel, caracterizados por estar enfocados a la resolución de problemas en campos de aplicación específicos y los programas escritos en ellos ser fácilmente trasladables de una a otra computadora.
b) Lenguajes ensambladores y máquina, totalmente adaptados y predeterminados por la CPU de la máquina.

1.4.2. Lenguajes declarativos.

Están basados en la definición de funciones o relaciones. No utilizan instrucciones de asignación (sus variables no almacenan valores). Son los más fáciles de utilizar (no se requieren conocimientos específicos de informática), están muy próximos al hombre. Se suelen denominar también lenguajes de órdenes, ya que los programas están formados por sentencias que ordenan "qué es lo que se quiere hacer", no teniendo el programador que indicar a la computadora el proceso detallado (el algoritmo) de cómo hacerlo".
En este grupo se incluyen ciertos lenguajes especializados en funciones tales como recuperación de la información en bases de datos (NATURAL e IMS), análisis de circuitos electrónicos (SPICE), y realización de cálculos estadísticos (BMDP, SPSS, SAS, etc.).

Se dividen en lenguajes funcionales y lógicos.

1.4.2.1. Lenguajes funcionales.

Los lenguajes funcionales son un tipo de lenguajes declarativos, en los que los programas están formados por una serie de definiciones de funciones. Ejemplos de estos lenguajes son el LISP y el SCHEME. Se suelen aplicar a problemas de Inteligencia Artificial.

1.4.2.2. Lenguajes lógicos.

Los lenguajes lógicos son el otro tipo de lenguajes declarativos, y en ellos los programas están formados por una serie de definiciones de predicados. También se les denomina lenguajes de programación lógica, y el mayor exponente es el lenguaje PROLOG. Se aplican sobre todo en la resolución de problemas de Inteligencia Artificial.

¿Qué es un algoritmo?

En matemáticas, ciencias de la computación y disciplinas relacionadas, un algoritmo (del griego y latin, dixit algorithmus y éste a su vez del matemático persa Al Juarismi) es un conjunto preescrito de instrucciones o reglas bien definidas, ordenadas y finitas que permite realizar una actividad mediante pasos sucesivos que no generen dudas a quien deba realizar dicha actividad. Dados un estado inicial y una entrada, siguiendo los pasos sucesivos se llega a un estado final y se obtiene una solución. Los algoritmos son el objeto de estudio de la algoritmia.

algoritmos son independientes tanto del lenguaje de programación en que se expresan como de la computadora que los ejecuta. En cada problema el algoritmo se puede expresar en un lenguaje diferente de programación y ejecutarse en una computadora distinta.

Características de los algoritmos
Las características fundamentales que debe cumplir todo algoritmo son:
· Un algoritmo debe ser preciso e indicar el orden de realización de cada paso.
· Un algoritmo debe estar definido. Si se sigue un algoritmo dos veces, se debe obtener el mismo resultado cada vez.
· Un algoritmo debe ser finito. Si se sigue un algoritmo, se debe terminar en algún momento; o sea debe de tener un número finito de pasos.

LOS ALGORITMOS SE CLASIFICAN EN:

1.-Ordenamiento Burbuja

Este es el algoritmo más sencillo probablemente. Ideal para empezar. Consiste en ciclar repetidamente a través de la lista, comparando elementos adyacentes de dos en dos. Si un elemento es mayor que el que está en la siguiente posición se intercambian.
Ventajas:
· Fácil implementación.
· No requiere memoria adicional.
Desventajas:
· Muy lento.
· Realiza numerosas comparaciones.
· Realiza numerosos intercambios.

2.-Ordenamiento por Selección

Este algoritmo también es sencillo. Consiste en lo siguiente:
· Buscas el elemento más pequeño de la lista.
· Lo intercambias con el elemento ubicado en la primera posición de la lista.
· Buscas el segundo elemento más pequeño de la lista.
· Lo intercambias con el elemento que ocupa la segunda posición en la lista.
· Repites este proceso hasta que hayas ordenado toda la lista.
Ventajas:
· Fácil implementación.
· No requiere memoria adicional.

· Realiza pocos intercambios.
· Rendimiento constante: poca diferencia entre el peor y el mejor caso.
Desventajas:
· Lento.
· Realiza numerosas comparaciones.

3.-Ordenamiento por Inserción

En este tipo de algoritmo los elementos que van a ser ordenados son considerados uno a la vez. Cada elemento es INSERTADO en la posición apropiada con respecto al resto de los elementos ya ordenados.

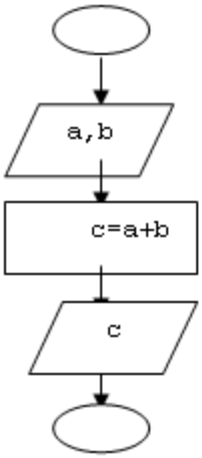
Ventajas:
· Fácil implementación.
· Requerimientos mínimos de memoria.
Desventajas:
· Lento.
· Realiza numerosas comparaciones.
Este también es un algoritmo lento, pero puede ser de utilidad para listas que están ordenadas o semiordenadas, porque en ese caso realiza muy pocos desplazamientos.

4.-Ordenamiento Rápido

Esta es probablemente la técnica más rápida conocida. Fue desarrollada por C.A.R. Hare en 1960. El algoritmo original es recursivo, pero se utilizan versiones interactivas para mejorar su rendimiento (los algoritmos recursivos son en general más lentos que los iterativos, y consumen más recursos). El algoritmo fundamental es el siguiente:
· Eliges un elemento de la lista.
· Buscas la posición que le corresponde en la lista ordenada.
· Acomodas los elementos de la lista a cada lado del elemento de división, de manera que a un lado queden todos los menores que él y al otro los mayores . En este momento el elemento de división separa la lista en dos sublistas.
· Realizas esto de forma recursiva para cada sublista mientras éstas tengan un largo mayor que 1. Una vez terminado este proceso todos los elementos estarán ordenados.
Una idea preliminar para ubicar el elemento de división en su posición final sería contar la cantidad de elementos menores y colocarlo un lugar más arriba. Pero luego habría que mover todos estos elementos a la izquierda del elemento, para que se cumpla la condición y pueda aplicarse la recursividad. Reflexionando un poco más se obtiene un procedimiento mucho más efectivo. Se utilizan dos índices: i, al que llamaremos contador por la izquierda, y j, al que llamaremos contador por la derecha. El algoritmo es éste:
· Recorres la lista simultáneamente con i y j: por la izquierda con i (desde el primer elemento), y por la derecha con j (desde el último elemento).
· Cuando lista sea mayor que el elemento de división y lista[j] sea menor los intercambias.
· Repites esto hasta que se crucen los índices.
· El punto en que se cruzan los índices es la posición adecuada para colocar el elemento de división, porque sabemos que a un lado los elementos son todos menores y al otro son todos mayores (o habrían sido intercambiados).
Al finalizar este procedimiento el elemento de división queda en una posición en que todos los elementos a su izquierda son menores que él, y los que están a su derecha son mayores.
Ventajas:
· Muy rápido
· No requiere memoria adicional.
Desventajas:
· Implementación un poco más complicada.

· Recursividad (utiliza muchos recursos).
· Mucha diferencia entre el peor y el mejor caso.

Ejemplos de los algoritmos



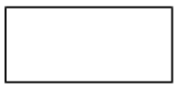
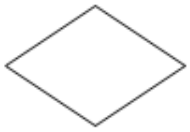
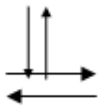
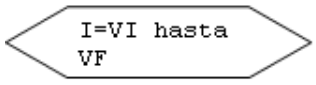
Inicio

solicita dos valores y los guarda en a y b

suma el contenido de a y b y lo almacena en c

muestra el resultado almacenado en la variable c

fin

SÍMBOLO	***DESCRIPCIÓN***
	Indica el inicio y el final de nuestro diagrama de flujo.
	Indica la entrada y salida de datos.
	Símbolo de proceso y nos indica la asignación de un valor en la memoria y/o la ejecución de una operación aritmética.
	Símbolo de decisión indica la realización de una comparación de valores.
	Líneas de flujo o dirección. Indican la secuencia en que se realizan las operaciones.
	Repetitiva Desde