INSTITUTO BARRANQUERAS

Lenguajes de Programación

Para que las computadoras, teléfonos móviles, tabletas y otros dispositivos electrónicos funcionen, se requiere un lenguaje de programación que les indique los comandos y permite completar diferentes actividades.

¿Qué es la programación?

La programación es el acto de programar, es decir, organizar una secuencia de pasos ordenados a seguir para hacer cierta cosa. Es común hablar de programación a la hora de organizar una salida, las vacaciones o la lista de programas con sus días y horarios de emisión de los canales de televisión o la lista de películas de un cine.

En el ámbito de la informática, la programación refiere a la acción de crear programas o aplicaciones a través del desarrollo de un código fuente, que se basa en el conjunto de instrucciones que sigue la computadora para ejecutar un programa. La programación es lo que permite que una computadora funcione y realice las tareas que el usuario solicita.

Código fuente: Es el conjunto de líneas de texto que expresan, en un lenguaje de programación determinado, los pasos que debe seguir la computadora para la correcta ejecución de un programa específico. Es decir, se trata de las instrucciones que el programador del software compiló para que pudieran ser transmitidas a un sistema.

La programación es un procedimiento mediante el cual podemos procesar datos y obtener un resultado, por ejemplo, como en una operación matemática 5+1=6, ¡así de simple! Tenemos instrucciones, un procesamiento y un resultado.

Y entonces ¿qué es un lenguaje de programación?

Es un lenguaje formal que, mediante una serie de instrucciones, le permite a un programador escribir un conjunto de órdenes, acciones consecutivas, datos y algoritmos para, de esa forma, crear programas que controlen el comportamiento físico y lógico de una máquina.

Algoritmo: un algoritmo es una serie de instrucciones secuenciales —es decir, que van uno después del otro— que permiten ejecutar acciones.

En informática, se conoce como lenguaje de programación a un programa destinado a la construcción de otros programas informáticos. Su nombre se debe a que comprende un lenguaje formal que está diseñado para organizar algoritmos y procesos lógicos que serán

procesados por un sistema informático, permitiendo controlar así su comportamiento físico, lógico y su comunicación con el usuario humano.

Dicho lenguaje está compuesto por símbolos y reglas sintácticas y semánticas, expresadas en forma de instrucciones y relaciones lógicas, mediante las cuales se construye el código fuente de una aplicación.

El lenguaje de programación es la base para construir todas las aplicaciones digitales que se utilizan en el día a día y se clasifican en dos tipos principales: lenguaje de bajo nivel y de alto nivel.

Las computadoras sólo entienden un lenguaje conocido como código binario o código máquina, consistente en ceros y unos. Es decir, sólo utiliza 0 y 1 para codificar cualquier acción.

Lenguaje de máquina son las instrucciones que entiende la computadora (el procesador para ser más exactos) en código binario (unos y ceros).

Programación según su nivel

Los lenguajes más próximos a la arquitectura hardware se denominan lenguajes de bajo nivel y los que se encuentran más cercanos a los programadores y usuarios se denominan lenguajes de alto nivel.

Lenguajes de bajo nivel

Son lenguajes totalmente dependientes de la máquina, es decir que el programa que se realiza con este tipo de lenguajes no se pueden migrar o utilizar en otras maquinas.

Al estar prácticamente diseñados a medida del hardware, aprovechan al máximo las características del mismo.

Dentro de este grupo se encuentran:

- El lenguaje maquina: este lenguaje ordena a la máquina las operaciones fundamentales para su funcionamiento. Consiste en la combinación de 0's y 1's para formar las ordenes entendibles por el hardware de la computadora. Este lenguaje es mucho más rápido que los lenguajes de alto nivel.

- El lenguaje ensamblador es un derivado del lenguaje máquina y está formado por abreviaturas de letras y números llamadas mnemotécnicos.

Lenguajes de alto nivel

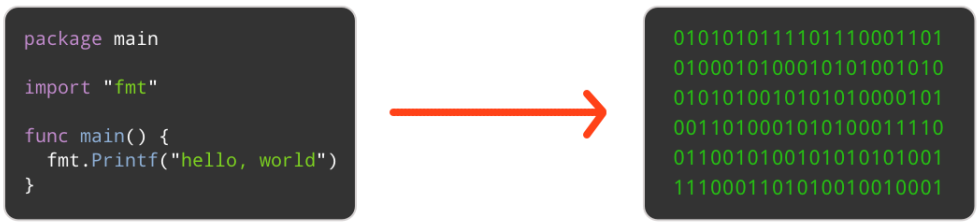
Son aquellos que se encuentran más cercanos al lenguaje natural que al lenguaje máquina. Se tratan de lenguajes independientes de la arquitectura de la computadora. Por lo que, en principio, un programa escrito en un lenguaje de alto nivel, lo puedes migrar de una máquina a otra sin ningún tipo de problema.

Los lenguajes de programación de alto nivel son lenguajes en los cuales las instrucciones que se envían para que la computadora ejecute algunas órdenes son parecidas al lenguaje humano.

El lenguaje de programación de alto nivel usa palabras similares al inglés, así como símbolos, signos de puntuación y aritméticos de manera que permite el desarrollo de programas.

Un programa que está escrito en lenguaje de alto nivel se le denomina como programa fuente, en el cual como una computadora no puede entender cierto programa, necesita la ayuda de un compilador que tiene la tarea traducirlo a lenguaje máquina. Comúnmente, los programas que son traducidos a lenguaje máquina por los compiladores se enlazan con algunos otros códigos o programas de bibliotecas del lenguaje de programación específico y esto hace que se conviertan en un archivo ejecutable para la máquina.

Estos lenguajes permiten al programador olvidarse por completo del funcionamiento interno de la maquina/s para la que están diseñando el programa.



Lenguaje de alto nivel que entiende el programador

Lenguaje de máquina que entiende el procesador

Programación según su compilación

Para que las persona se entienda deben hablar el mismo idioma, lo mismo pasa con las computadoras, para que puedan darle instrucciones deben poder entender, para esto existen los compiladores, que son los encargados de traducir el lenguaje humano al lenguaje (o código) de una computadora.

Lenguaje Compilado

Un lenguaje compilado es aquel cuyo código fuente, escrito en un lenguaje de alto nivel, es traducido por un compilador a un archivo ejecutable entendible para la máquina. Con ese archivo se puede ejecutar el programa cuantas veces sea necesario.

En este tipo de lenguaje el código se compila, ¿para qué? para crear un paquete de código máquina (código binario), así la computadora puede ejecutar las instrucciones.

A los lenguajes compilados los vemos más en software de escritorio ya que requieren de mayores recursos y de acceso a archivos determinados.

Lenguaje Interpretado

Este lenguaje ¡ya no cuenta con un compilador! El código va directo a la máquina quien ahora tiene un intérprete, que traduce el código y lo convierte a su lenguaje, entonces ¿Un compilador es lo mismo que un intérprete?, bueno, digamos que tienen la misma funcionalidad (traducir), pero su diferencia radica en que el intérprete lo realiza al momento de ejecución (cuando lo solicitas) y al ser en tiempo real puede alentar el proceso.

La principal diferencia entre un lenguaje compilado y uno interpretado es que el lenguaje compilado requiere un paso adicional antes de ser ejecutado, la compilación, que convierte el código que escribes a lenguaje de máquina. Un lenguaje interpretado, por otro lado, es convertido a lenguaje de máquina a medida que es ejecutado.

Ventajas y desventajas:

En general, el ciclo de desarrollo (el tiempo entre el momento en que escribes el código y lo pruebas) es más rápido en un lenguaje interpretado. Eso se debe a que en lenguajes compilados es necesario realizar el proceso de compilación cada vez que cambias el código fuente, aunque con herramientas adicionales se puede automatizar.

Otra desventaja de un lenguaje compilado es que cuando compilas un programa debes crear ejecutables para cada uno de los sistemas operativos en los que lo vayas a utilizar. Un ejecutable creado para Linux no va a servir en Windows por ejemplo.

Sin embargo, un lenguaje compilado es mucho más rápido que uno interpretado. Esto se debe a que cuando es ejecutado ya se encuentra en código de máquina y eso también le permite hacer algunas optimizaciones que no son posibles con un lenguaje interpretado.

Además de la velocidad, otra desventaja de un lenguaje interpretado es que, para ser ejecutado, debes tener instalado el interpretador. Esto no es necesario en un lenguaje compilado que es convertido a lenguaje de máquina.

Programación según su propósito

Lenguajes de propósito general

Son lenguajes que pueden ser usados para varios propósitos, acceso a bases de datos, comunicación entre computadoras, comunicación entre dispositivos, captura de datos, cálculos matemáticos, diseño de imágenes o páginas, etc.

Lenguajes de propósito específico

Es un lenguaje de programación dedicado a resolver un problema en particular, representar un problema específico y proveer una técnica para solucionar una situación particular.

Están hechos con un propósito concreto con la finalidad de resolver problemas determinados, ya sean problemas estadísticos, programación de máquinas, para simulación de sistemas, entre otros.

Programación según su Tipado

El tipado se refiere a cómo declaramos los tipos de variables. Por ejemplo, algunas las declaramos como enteras, algunas otras como cadena, flotantes, etcétera. Y en algunos lenguajes, no necesitamos declarar el tipo.

Por otro lado, el tipado fuerte no permite hacer operaciones entre objetos de distintos tipos. No podemos sumar una cadena más un entero. En cambio, en los débilmente tipado sí.

Tipado Fuerte

Aquí es en donde indicamos el tipo de dato al declarar la variable. Dicho tipo no puede ser cambiado nunca. Y no podemos operar entre distintos tipos.

Ventajas

Código expresivo: ahora sí sabremos de qué tipo espera un argumento una función

Menos errores: Nos olvidaremos de ver el tipo de variable antes de hacer operaciones

Desventajas

Escribir más código: tenemos que declarar el tipo de variable al declararla

Tipado Débil

La mayoría de veces, el tipado débil es en donde no indicamos el tipo de variable al declararla. La verdadera diferencia es que podemos asignar, por ejemplo, un valor entero a una variable que anteriormente tenía una cadena.

También podemos operar aritméticamente con variables de distintos tipos. Por ejemplo, sumar “x” + 5.

Ventajas

Nos olvidamos de declarar el tipo

Podemos cambiar el tipo de la variable sobre la marcha.

Escribimos menos código

Desventajas

Al hacer operaciones, a veces éstas salen mal. Por ejemplo, puede que intentemos sumar 500 + “400.00” + 10, cosa que será errónea

Inseguridad: existe la posibilidad de que un atacante descubra una vulnerabilidad en donde nosotros esperemos una variable de determinado tipo pero se reciba otra.