7 Lenguajes de programación

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.

Es el lenguaje mediante el que se comunican programador y máquina.

Se pueden clasificar según diversos criterios, algunos de ellos son los siguientes:

- Según su **proximidad al lenguaje máquina**
    - lenguajes de bajo nivel
    - lenguajes de alto nivel.
- Según su **evolución histórica**
    - Primera generación
    - Segunda generación
    - Tercera generación
    - Cuarta generación
    - Quita generación
- Según su **paradigma** de programación

7.1 Clasificación según su proximidad al lenguaje máquina

7.1.1 Lenguajes de programación de bajo nivel

Son lenguajes totalmente orientados a la máquina. Son el tipo de lenguajes que utilizan los ordenadores para realizar operaciones del sistema. Un código escrito en un lenguaje de bajo nivel interactúa directamente con el procesador de la computadora o CPU y es capaz de ejecutar comandos muy básicos que son generalmente difíciles de leer por una persona.

- **Lenguaje máquina.** Es el más primitivo de los lenguajes y es una colección de dígitos binarios o bits (0 y 1) que la computadora lee e interpreta y son los únicos idiomas que las computadoras entienden.
- **Lenguaje ensamblador.** El lenguaje ensamblador es el primer intento de sustitución del lenguaje de máquina por uno más cercano al utilizado por los humanos. Un programa escrito en este lenguaje es almacenado como texto (tal como programas de alto nivel) y consiste en una serie de instrucciones que corresponden al flujo de órdenes ejecutables por un microprocesador. Sin embargo, dichas máquinas no comprenden el lenguaje ensamblador, por lo que

se debe convertir a lenguaje máquina mediante un programa llamado Ensamblador. Ejemplo: Assembly.

7.1.2 Lenguajes de programación de alto nivel

Tienen como objetivo facilitar el trabajo del programador, ya que utilizan unas instrucciones más fáciles de entender.

Además, el lenguaje de alto nivel permite escribir códigos mediante idiomas que conocemos (español, inglés, etc.) y luego, para ser ejecutados, se traduce al lenguaje de máquina mediante traductores o compiladores.

- **Traductor**. Traducen programas escritos en un lenguaje de programación al lenguaje máquina de la computadora y a medida que va siendo traducida, se ejecuta.
- **Compilador**. Permite traducir todo un programa de una sola vez, haciendo una ejecución más rápida y puede almacenarse para usarse luego sin volver a hacer la traducción.

En otras palabras, el lenguaje de bajo nivel es cercano a los idiomas de las máquinas mientras que el lenguaje de alto nivel está más cerca del entendimiento e idioma humano.

Los lenguajes de programación de alto nivel deben convertirse a lenguajes de programación de bajo nivel utilizando un intérprete o compilador, según lo necesite el lenguaje.

Lenguaje interpretado

Es aquel en el cual sus instrucciones o más bien el código fuente, escrito por el programador en un lenguaje de alto nivel, es traducido por el intérprete a un lenguaje entendible para la máquina paso a paso, instrucción por instrucción. El proceso se repite cada vez que se ejecuta el programa el código en cuestión. Ejemplos Ruby, Python, PHP.

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 en determinada plataforma. Con ese archivo se puede ejecutar el programa cuantas veces sea necesario sin tener que repetir el proceso por lo que el tiempo de espera entre ejecución y ejecución es ínfimo. Ejemplos C++, Fortran, Visual Basic.

7.2 Clasificación según su evolución histórica

7.2.1 Primera generación 1GL

Conocido como código máquina. Cadenas interminables de secuencias de números 1 y 0 que conforma operaciones que la máquina puede entender sin interpretación alguna.

Características:

- El lenguaje está ligado íntimamente a la CPU, baja portabilidad.
- No existen los comentarios
- Los datos se referencian por medio de las direcciones de memoria donde se encuentran.
- Las instrucciones realizan operaciones muy simples.
- Poca versatilidad para la redacción de instrucciones, ya que tiene un formato muy rígido.

7.2.2 Segunda generación 2GL

Constituye el primer intento de sustitución del lenguaje máquina por uno más cercano al usado por los humanos. Este acercamiento se plasma en las siguientes aportaciones:

- Uso de una notación simbólica o nemotécnica para representar los códigos de operación. Normalmente mediante abreviaturas en ingles por ejemplo ADD.
- Direccionamiento simbólico. En lugar de utilizar direcciones binarias absolutas, los datos pueden identificarse con nombres. (Variables).
- Se permite el uso de comentarios entre las líneas de instrucciones,

También son lenguajes ensambladores de bajo nivel.

Aparte de esto, el lenguaje ensamblador presenta la mayoría de los inconvenientes del lenguaje máquina:

- Repertorio muy reducido y rígido de instrucciones.
- Baja portabilidad y la fuerte dependencia del hardware.
- Mantiene la ventaja del uso óptimo de los recursos hardware, permitiendo la obtención de un código muy eficiente.

7.2.3 Tercera generación 3GL

Los esfuerzos encaminados a hacer la labor de programación independiente de la máquina dieron como resultado la aparición de los lenguajes de programación de alto nivel. Estos lenguajes, más evolucionados, utilizan palabras y frases relativamente fáciles de entender y proporcionan también facilidades para expresar alteraciones del flujo de control de una forma bastante sencilla e intuitiva. Ejemplo: Fortran, C, C++, Java, BASIC …

Características:

- Son independientes de la arquitectura física del ordenador. Portabilidad.
- Una sentencia es traducida a varias instrucciones en lenguaje máquina.

- Las operaciones se expresan con sentencias muy parecidas al lenguaje matemático o al lenguaje natural. Se utilizan, por lo general, palabras o términos en inglés.
- Permiten el uso de comentarios.
- Hacen uso de variables.

7.2.4 Cuarta generación 4GL

El objetivo de los lenguajes de programación de cuarta generación es la de proveer de un nivel mayor de abstracción al programa del hardware del equipo.

Estos lenguajes proporcionan:

- Manejo de base de datos.
- Desarrollo de interfaces gráficas.
- Desarrollo de aplicaciones web.
- Generación de informes.
- Optimización matemática.

Ejemplos: SQL, MATLAB, Scilab

7.2.5 Quinta generación 5GL

Un lenguaje de programación de quinta generación es un lenguaje de programación basado en la resolución de problemas utilizando restricciones dadas al programa, en lugar de utilizar un algoritmo escrito por un programador.

Lenguajes de quinta generación se utilizan principalmente en la investigación de la inteligencia artificial. Prolog, OPS5, y Mercury son ejemplos de lenguajes de quinta generación. Estos tipos de lenguajes también se basaron en Lisp (List Processor).

7.3 Clasificación según su paradigma de programación

Un paradigma de programación es una manera o estilo de programación de software. Existen diferentes formas de diseñar un lenguaje de programación y varios modos de trabajar para obtener los resultados que necesitan los programadores. Se trata de un conjunto de métodos sistemáticos aplicables en todos los niveles del diseño de programas para resolver problemas computacionales.

Los lenguajes de programación adoptan uno o varios paradigmas en función del tipo de órdenes que permiten implementar.

Un paradigma está delimitado en el tiempo en cuanto a aceptación y uso, porque nuevos paradigmas aportan nuevas o mejores soluciones que lo sustituyen parcial o totalmente. El paradigma actualmente más utilizado es la "orientación a objetos".

En general, la mayoría de los paradigmas son variantes de los dos tipos principales de programación, imperativa y declarativa. En la **programación imperativa** se describe paso a paso un conjunto de instrucciones que deben ejecutarse para variar el estado del programa y hallar la solución, es decir, un algoritmo en el que se describen los pasos necesarios para solucionar el problema. En la **programación declarativa** las sentencias que se utilizan lo que hacen es describir el problema que se quiere solucionar, se programa diciendo lo que se quiere resolver a nivel de usuario, pero no las instrucciones necesarias para solucionarlo. Esto último se realizará mediante mecanismos internos de inferencia de información a partir de la descripción realizada.

Práctica 1.3:

Desarrolla para exponer al menos los siguientes paradigmas de programación:

• Estructurado • Modular • Orientado a objetos • Funcional • Reactivo • Multiparadigma	➢ Características ➢ Lenguajes que los utilizan