Anexo web 3.1

Existe multitud de lenguajes de programación y la gran mayoría ha sufrido su propia evolución a lo largo del tiempo. No todos los lenguajes tienen las mismas características ni han sido creados para el mismo propósito y, por ello, se les puede clasificar según varios aspectos: fecha de creación, propósito, tipo de ejecución de su código, nivel de abstracción, lugar donde se ejecutan…

Clasificación según generación

En una primera clasificación se pueden agrupar los lenguajes según la generación a la que pertenezcan. Las 2 primeras generaciones son denominadas de *bajo nivel*, ya que existe una fuerte dependencia con el equipo físico sobre el que finalmente se desea programar.

- *Primera generación*:
  - Son el nivel más bajo sobre el que se podía programar.
  - Conocidos como *lenguaje máquina*.
  - Utilizan código binario (basado en 0's y 1's).
  - Cada equipo tenía su propio código y de ahí que se denominase código máquina.

- *Segunda generación*:
  - Basados en lenguajes simbólicos.
  - Los códigos empleados finalmente se traducen a código máquina.
  - Aparecieron los primeros lenguajes ensambladores.
  - Cada equipo tenía su propio lenguaje ensamblador con un conjunto reducido de instrucciones.

- *Tercera generación*:
  - Comprende los lenguajes de *alto nivel*.
  - Son muchos los lenguajes de programación que podríamos enumerar. Algunos de los lenguajes más populares son Fortran, Pascal, C, Cobol, Java …
  - Supusieron un gran avance en la elaboración de programas al desacoplar el lenguaje empleado de la arquitectura hardware sobre la que podría ejecutarse, con lo que el mismo programa podría ahora ser ejecutado en equipos de diferente arquitectura física.

- *Cuarta generación*:
  - Lenguajes en los que se especifica qué resultado se quiere obtener y no el cómo.
  - Un ejemplo típico es el lenguaje estándar de consulta de bases de datos, más conocido como SQL.

- *Quinta generación*:
  - Son lenguajes generalmente orientados a inteligencia artificial.
  - Ejemplos de estos lenguajes son LISP (utiliza listas) y PROLOG (utiliza predicados).

SABÍAS QUE...

El lenguaje ensamblador utiliza mnemónicos, que son agrupaciones de caracteres alfanuméricos que representan las órdenes a realizar. La traducción de mnemónicos a código máquina entendible por el microcontrolador se realiza con un programa ensamblador.

WWW

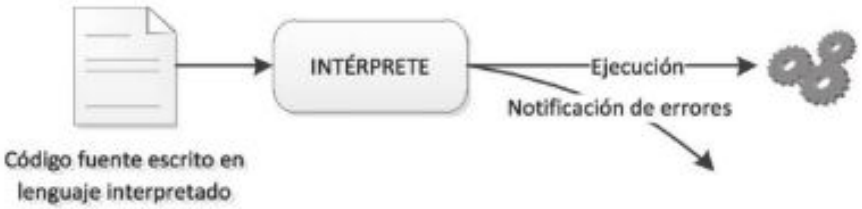
Recurso web

En la URL http://www.easy68k.com/ podrás acceder al proyecto *EASy68K* con el que podrás editar, ensamblar y ejecutar programas para el microcontrolador Motorola 68000.

Clasificación según su tipo de ejecución

También puede llevarse a cabo una clasificación en dos grandes categorías dependiendo de cómo se lleve a cabo su ejecución:

✓ *Lenguajes interpretados* son aquellos que necesitan de un intérprete para realizar la traducción del código fuente a código objeto. El intérprete ejecuta el código fuente que recibe según lo va leyendo (no hay fases previas):



Ejemplos de este tipo de lenguajes son: PHP, Perl, Lisp...

✓ *Lenguajes compilados* son aquellos que necesitan de un compilador para realizar la traducción del código fuente a código objeto. El compilador recibe el código fuente, lo analiza, detecta errores, realiza las correcciones oportunas, optimiza el código y, en caso de que todo esté correcto, genera un código objeto que almacena en un archivo independiente.

Además de compiladores suelen emplearse enlazadores (*linkers*) y depuradores (*debuggers*).

La ejecución de un programa compilado es más rápida que la de un programa interpretado. Sin embargo los programas interpretados facilitan la portabilidad entre sistemas a la vez que ocupan menos tamaño.

INVESTIGA

Indaga sobre los siguientes lenguajes de programación y clasifícalos en *interpretados* o *compilados:* Pascal, C, Lisp, Lips, Ruby, Python, Delphi, C++, Eiffel, Smaltalk, PHP y Fortran.

Clasificación según su propósito

No todos los lenguajes han sido creados para atender los mismos problemas ni para ser empleados en todos los ámbitos. Algunos lenguajes son muy polivalentes y permiten abarcar un gran espectro de escenarios mientras que otros tienen un objetivo muy específico:

- *Propósito general*: este tipo de lenguajes son muy polivalentes y se adaptan con facilidad a cualquier escenario. Ejemplos de este tipo de lenguajes son: Visual Basic, Java, Delphi…
- *Propósito específico*: estos lenguajes son utilizados en escenarios muy concretos y para implementar algoritmos muy específicos. Ejemplos de este tipo de lenguajes son: MATLAB (operaciones matemáticas), ADA (sistemas en tiempo real), LISP (inteligencia artificial), CSound (creación de sonido), POV-Ray (creación de gráficos), SCRATCH (uso educativo)…

INVESTIGA

Los sistemas de tiempo real son desarrollados con lenguajes de tiempo de real que permiten realizar programas concurrentes, con comunicación y sincronización entre procesos e hilos, tolerantes a errores y que gestionan el tiempo de forma muy precisa. Investiga sobre los lenguajes que se pueden utilizar para desarrollar este tipo de sistemas.