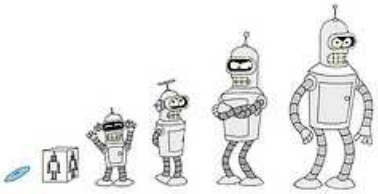
Programación: Lenguajes de programación

Img 0-A. Evolución del hombre

Imagen de Flickrcc

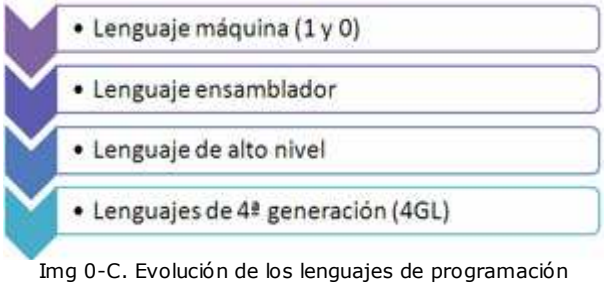
La evolución del hombre. Todos la hemos estudiado alguna vez y conocemos los principales cambios que se han producido a lo largo de la historia.



Img 0-B. Evolución de las máquinas

Imagen de Flickrcc

De forma pareja se ha producido la evolución de los útiles, las herramientas, las máquinas y los procesos de fabricación. Los grandes cambios que han marcado las etapas de la evolución han sido generados por nuevos inventos, descubrimientos o avances tecnológicos.



Img 0-C. Evolución de los lenguajes de programación

Imagen de producción propia

La evolución de los ordenadores se ha basado en avances tecnológicos y ha ido acompañada de cambios en los lenguajes de programación. En principio, se trataba de "hablar" con el ordenador en su idioma, es decir, en el lenguaje máquina basado en el sistema binario (0,1).

Pero apareció una "intrusa" en la línea del tiempo: ADA LOVELACE. Gracias a ella se sentaron las bases para conseguir hablar con el ordenador en lenguaje natural. Desde entonces, esta ha sido la meta de los lenguajes de programación.

1 Programas y algoritmos

Un **programa** es un conjunto de instrucciones ordenadas secuencialmente que permiten a un ordenador interpretar una información de entrada, procesarla y producir una información de salida.

Un **algoritmo** es una secuencia definida, ordenada y finita de instrucciones que permiten hallar la solución a un problema.

Para crear un programa se utiliza un lenguaje de programación, generando lo que se llama programa fuente. Pero este conjunto de instrucciones no tendrían sentido si el ordenador no pudiese entenderlas, por lo que es necesario convertirlo en el programa objeto.

Para traducir el programa fuente a programa objeto se utilizan diferentes herramientas informáticas, que pueden generar dos tipos de programas:

- Programa compilado.- el programa se genera como un todo y una vez obtenido el programa objeto, ya no hay que compilarlo a no ser que haya una modificación en el programa fuente.
- Programa interpretado.- el programa se ejecuta línea a línea y cada vez que se quiere ejecutar preciso volver a interpretarlo.

Los programas están formados por algoritmos y por la estructura de datos. Normalmente están divididos en módulos de modo que la complejidad de cada parte sea menor que la del programa completo, facilitando así el desarrollo del programa. Esta es la base de la programación estructurada que estudiarás en profundidad en el siguiente tema de esta unidad.

Img 1. Código fuente de un programa

Imagen de Wikipedia con licencia Creative Commons

Autoevaluación

Repasa las definiciones anteriores y elige la opción correcta.

1) Un conjunto de instrucciones ordenadas secuencialmente se llama:

a) Programa fuente	b) Programa objeto	c) Programa	☐

2) Cuando el programa fuente que se traduce como un todo se llama:

a) Compilado	b) Interpretado	c) Algoritmos	☐

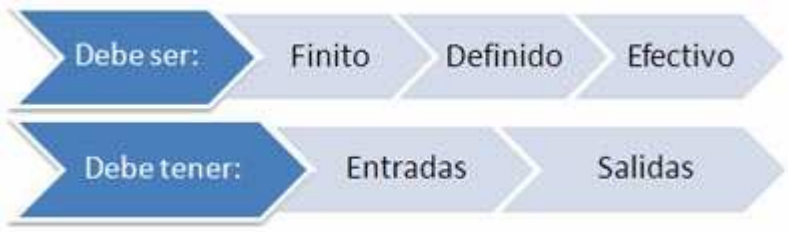
3) Cuando el programa fuente que se traduce línea a línea se llama:

a) Compilado	b) Interpretado	c) Ejecutable	☐

Comprobar

Para construir un edificio, hay que empezar por los cimientos. Esto se puede aplicar también a la programación. A lo largo del tema, aprenderás cómo se crea un programa y un algoritmo. Pero antes de llegar a este punto, es necesario establecer una serie de normas o condiciones que debe cumplir el diseño de un algoritmo para que su desarrollo posterior, y por tanto el del programa que genere, sea correcto.

Las siguiente condiciones son los "cimientos" de los algoritmos:



Autoevaluación

Ya te has dado cuenta de que no hemos explicado estas condiciones. Como son sencillas, seguro que eres capaz de indicar a qué se refiere cada una de ellas.

- Debe cumplir las especificaciones para las que fue creado.	
- Debe tener una o varias entradas que puedan ser interpretadas por el ordenador.	
- Debe poder realizarse en un número concreto de pasos.	
- Debe definirse de forma precisa, evitando toda ambigüedad.	
- Debe tener una o varias salidas relacionadas con las entradas.	

Comprobar

1.1 Programación

Importante

La **programación** es el proceso por el cual se escribe el código fuente de un programa. También se incluyen en este proceso los procesos de análisis, prueba y redefinición de los programas.

La persona que se encarga de realizar el programa se llama **programador** o **desarrollador de software**.

La programación se basa en una serie de factores los cuales se pueden interpretar como normas a seguir por los programadores. Éstos son los siguientes:

Autoevaluación

Imagina que eres programador y tienes que diseñar un programa que consiste en "ir al cajero automático a sacar dinero". Te proponemos las siguientes opciones para resolver el problema, pero ¡ojo! en cada una de ellas falta algún factor fundamental en programación. Identifica cuál es.

a) Después de encontrar el cajero, la tarjeta no es válida en él.	
b) Cuando llego al cajero, está estropeado y no puedo sacar el dinero.	
c) Después de llegar al cajero e introducir la tarjeta, me pide una clave, después el tipo de moneda, después otra clave y al final me envía a otro cajero.	
d) Para ir al cajero que está a 1 Km, decido pasar primero a visitar a un amigo, tomar un café en su casa y volver en autobús.	

Comprobar

Para saber más

Muchas veces habrás oído hablar de la usabilidad de los programas. No se debe confundir este término con ninguno de los factores que hemos visto en la tabla anterior. La ISO ofrece dos definiciones de usabilidad:

ISO/IEC 9126	*"La usabilidad se refiere a la capacidad de un software de ser comprendido, aprendido, usado y ser atractivo para el usuario, en condiciones específicas de uso".*
ISO/IEC 9241	*"Usabilidad es la eficacia, eficiencia y satisfacción con la que un producto permite alcanzar objetivos específicos a usuarios específicos en un contexto de uso específico".*

La interpretación de la primera definición es que la usabilidad depende no sólo del producto sino también del usuario. La segunda, sin embargo, se centra en la calidad del uso que se hace del programa.

1.2 Paradigmas de la programación

En la evolución de la programación han surgido diversas técnicas de programación que se han ido adaptando a las necesidades tecnológicas e informáticas del momento. Aunque la forma de enfocar la elaboración de los programas es diferente en cada una de ellas, el objetivo es el mismo: facilitar la creación y el mantenimiento de programas informáticos. Estas técnicas se han traducido en diferentes filosofías de creación de programas que son los denominados paradigmas de programación.

Importante

Un **paradigma de programación** representa un conjunto de métodos sistemáticos aplicables en todos los niveles del diseño de programas. Es decir, un paradigma es una filosofía para la creación de programas.

Aunque hay muchos paradigmas de programación, en la siguiente tabla encontrarás los más comunes:

Paradigma imperativo ▸ Los programas imperativos contienen instrucciones que dicen al ordenador cómo realizar una tarea. Los primeros lenguajes imperativos fueron los códigos máquina de los ordenadores, que utilizaban instrucciones sencillas y permitían implementar el hardware fácilmente, pero no servían para desarrollar programas complejos. ▸ El primer lenguaje imperativo que posibilitó la creación de programas con un nivel de complejidad elevado fue FORTRAN. Hoy en día está representado por los lenguajes de programación BASIC, C ó PASCAL, entre otros.
Paradigma funcional ▸ Los programas funcionales se basan en el uso de una o más funciones dentro de las cuales se pueden utilizar funciones creadas anteriormente. Su objetivo es dividir el programa en módulos de forma que cada uno de éstos realice una única función. ▸ El primer lenguaje de programación funcional fue LISP. Existen dos tipos de lenguajes funcionales: los puros (como HASKELL) y los híbridos (SAP, ML, Scheme).
Paradigma lógico ▸ La programación lógica comprende la programación declarativa y la funcional. El proceso de elaboración de programas está basado en la lógica de primer orden y, a diferencia de los demás paradigmas, especifica qué debe hacer el programa y no cómo hacerlo. ▸ Se emplea en aplicaciones de inteligencia artificial. El lenguaje de programación lógica por excelencia es PROLOG.
Paradigma orientado a objetos (POO) ▸ La programación orientada a objetos expresa un programa como un conjunto de objetos, que colaboran entre ellos para realizar tareas. Esto permite hacer los programas y módulos más fáciles de escribir, mantener, reutilizar y volver a utilizar. Su uso se popularizó a principios de los 90 y actualmente son muchos los lenguajes de programación asociados a este paradigma. ▸ Las características del paradigma orientado a objetos son: encapsulamiento, abstracción, polimorfismo y herencia. ▸ Muchos lenguajes utilizados en la actualidad están orientados a objetos, como Java, C++, Python o Delphi. Un lenguaje completamente orientado a objetos es Smalltalk.

Autoevaluación

Ahora que ya conoces los paradigmas de programación, ¿serías capaz de indicar cuál es su principal característica?

- Basado en la lógica de primer orden, especifica qué debe hacer el programa y no cómo hacerlo.	
- Contienen instrucciones que dicen al ordenador cómo realizar una tarea.	
- Su objetivo es dividir el programa en módulos de forma que cada uno realice una función.	
- Expresa un programa como un conjunto de objetos que colaboran para realizar tareas.	

Comprobar

Ejercicio resuelto

Este último paradigma de programación es uno de los más utilizados en la actualidad. Para comprenderlo bien, es necesario definir el concepto de objeto.

Un **objeto** contiene toda la información que permite definirlo e identificarlo frente a otros objetos. Está formado por:

- Estados.- son las propiedades del objeto representadas por variables.
- Métodos.- son los comportamientos que el objeto es capaz de hacer.

En este ejercicio te proponemos que intentes definir un objeto: "una película". Haz una lista con los estados y los métodos que contendría dicho objeto.

Para saber más

En este apartado sólo hemos expuesto las características generales de los paradigmas de programación. En el siguiente documento, podrás encontrar ejemplos de cada modelo y ampliar los conocimientos sobre ellos.

Paradigmas de Programación

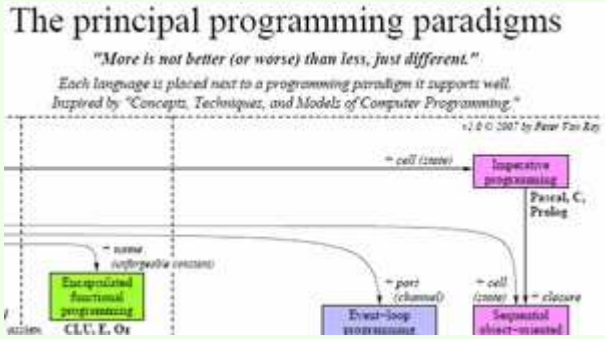
Curiosidad

Como en casi todos los campos, también existen mapas visuales de los paradigmas de programación. Te mostramos a continuación uno de ellos publicado en un blog sobre programación: Lambda the ultimate. En él se muestran los distintos paradigmas, las relaciones que hay entre ellos y los lenguajes de más representativos de cada uno de ellos.

La primera frase define perfectamente la intención de los paradigmas de programación. Aunque está en inglés, te la traducimos: "*Más no es mejor (o peor) que menos, sólo es diferente*".

Ya sabemos que en esta imagen no se puede leer nada, por eso te dejamos el enlace donde la podrás ver con precisión. Fíjate en los lenguajes de programación que aparecen porque serán los que estudies en el siguiente punto del tema.

Mapa visual de los paradigmas de programación (pdf)

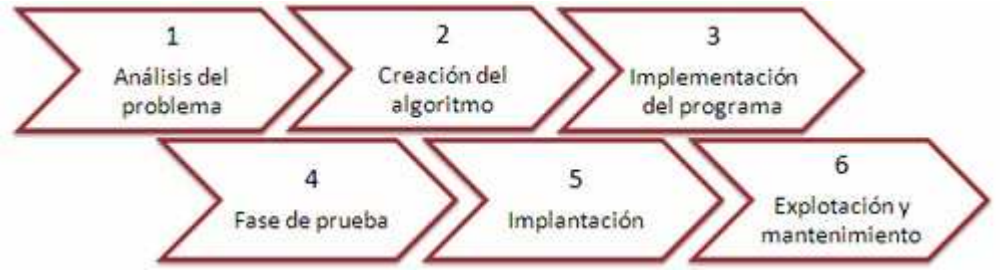


Img 2. Mapa de los paradigmas de programación

Imagen obtenida en la cofa con licencia Creative Commons

1.3 Creación de un programa

La creación de un programa informático es un proceso ordenado que ha de realizarse de modo secuencial. Desde el programa más sencillo, con pocas instrucciones, hasta la programación de grandes aplicaciones que contienen miles de líneas de instrucciones, se han de seguir una serie de pasos. Por supuesto, no son obligatorios, pero facilitan en gran medida dicho proceso.



Autoevaluación

Ya has visto las fases de creación de un programa, pero ¿serías capaz de determinar en que consite cada una de ellas?

Coloca el número correspondiente a cada fase.

- Implementar el programa en el lenguaje elegido siguiendo el algoritmo creado.	☐
- Recoger los requisitos del programa. Definir el proceso de automatización del problema.	☐
- Determinar fallos, mejoras o ampliaciones que los ususarios puedan necesitar.	☐
- Someter al programa a una serie de pruebas para examinar todas las opciones y posibilidades, detectando posibles errores.	☐
- Instalar el programa junto con los componentes necesarios (bases de datos, redes de comunicaciones, etc.).	☐
- Diseñar la arquitectura del programa. Crear el algoritmo que permita desarrollar el programa mediante diagramas de flujo o pseudocódigo.	☐

Comprobar

1.4 Creación de un algoritmo

En la creación de un programa, después del análisis del problema, se establecen las especificaciones del programa, es decir, qué debe hacer y cómo lo debe hacer. Si un algoritmo es correcto, es más fácil realizar la programación y se reduce la posibilidad de cometer errores.

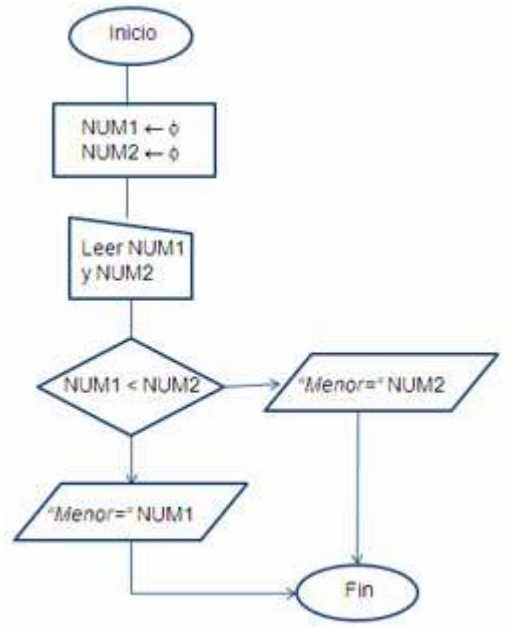
El desarrollo de un algoritmo se realiza en tres fases:

Los algoritmos pueden ser expresados de muchas maneras, destacando el lenguaje natural, los diagramas de flujo y el pseudocódigo. En la práctica, se utilizan los dos últimos ya que el lenguaje natural es más extenso.

Importante

- **Diagramas de flujo**.- permiten crear algoritmos mediante símbolos gráficos que representan operaciones específicas y que indican la secuencia de las operaciones mediante flechas. Están regidos por normas ISO.
- **Pseudocódigo**.- utilizan una sintaxis formada por frases o palabras en lenguaje común, instrucciones de programación y palabras clave que definen las estructuras básicas.

Para entenderlo mejor, te proponemos que estudies atentamente este ejemplo en el que se realiza el algoritmo de un programa que leerá dos número introducidos por el teclado y mostrará en pantalla el menor de los dos.




```
Inicio
        NUM1 ← ϕ
        NUM2 ← ϕ
Leer de teclado NUM1 y NUM2

SI NUM1 < NUM2
        Visualizar "Menor=" NUM1
SI NO
        Visualizar "Menor=" NUM2

Fin
```

Diagrama de flujo	Pseudocódigo

Autoevaluación

En la representación del algoritmo anterior, se aprecian las diferencias entre el pseudocódigo y el diagrama de flujo.

En la siguiente lista hemos puesto una serie de ventajas de un método sobre el otro, y viceversa. Para que te resulte fácil, sólo tienes que poner al lado de cada frase de qué método crees que es una ventaja, indicando DF, si es diagrama de flujo, o PS, si es pseudocódigo.

- Favorecen la comprensión del proceso ya que el cerebro humano reconoce fácilmente los dibujos.	☐
- Ocupan mucho menos espacio en el desarrollo del problema.	☐
- Permiten identificar y corregir errores de manera intuitiva en los procesos.	☐
- Se pueden observar los niveles en la estructura del programa fácilmente gracias a la indentación.	☐

Comprobar

Actividad de lectura

Ya te habrás dado cuenta de que aquí falta algo. A estas alturas del tema, todavía no te hemos explicado lo que significa cada uno de los símbolos utilizados para realizar un diagrama de flujo.

Te toca a ti averiguarlo. En las siguientes páginas web encontrarás toda la información que necesitas. Fíjate bien porque en la tarea del tema tendrás que realizar diagramas de flujo.

Organizadores: diagramas de flujo
Aprenda a crear diagramas de flujo

En el diagrama de flujo que aparece en este punto, puedes ver algunos de estos símbolos, ¿qué función tiene cada uno de ellos?

Para saber más

En el ejercicio anterior, has aprendido mucho sobre diagramas de flujo. En la siguiente página web encontrarás desarrollados algunos algoritmos importantes pero en pseudocódigo. Se trata de un proyecto de la Universitat Politécnica de Valencia en el que se ha creado el programa Grafo.
Son algoritmos complicados y no te vamos a pedir que los estudies, pero sí que te fijes en el pseudocódigo ya que entenderás mucho mejor los algoritmos.

Análisis de grafos
Un ejemplo: algoritmo de Dijkstra

Img 3. Análisis de grafos

Imagen de Grafos con licencia Creative Commons

2. Lenguajes de programación

Un **lenguaje de programación** es una herramienta que nos permite crear programas y software. Está formado por un conjunto de reglas sintácticas y semánticas y de símbolos que definen su estructura y el significado de sus elementos y expresiones.

¿Somos capaces de comunicarnos con el ordenador en su propio idioma? Puede que con un poco de tiempo... No, no podríamos realizar los programas "hablando" con 0 y 1. Los lenguajes de programación pretenden acercarse lo más posible al lenguaje humano o natural para comunicarnos con el ordenador en su propio idioma, es decir, en código máquina.

Existen muchos lenguajes de programación y también muchos criterios para clasificarlos. El que vamos a utilizar se basa en el criterio de abstracción, ya que es la clasificación más habitual. Por tanto, podemos tener dos tipos de lenguajes de programación:

- **Lenguajes de bajo nivel:** código máquina y ensamblador.
- **Lenguajes de alto nivel:** 1ª generación, 2ª generación, evolución de la 2ª generación, 3ª generación y 4ª generación.

Autoevaluación

Mira la presentación sobre la evolución de los lenguajes de programación y contesta las siguientes preguntas.

1) Los lenguajes primitivos eran orientados a:

- a) La máquina.
- b) Objetos.

2) COBOL es un lenguaje de programación creado para:

- a) Resolver ecuaciones algebraicas.
- b) Desarrollar aplicaciones comerciales.

3) MODULA-2, ADA y Delphi son lenguajes de programación que proceden de:

- a) BASIC
- b) PASCAL
- c) C

4) JAVA es un lenguaje de programación orientado a:

- a) La máquina.
- b) Objetos.

2.1 Lenguajes de bajo nivel

Importante

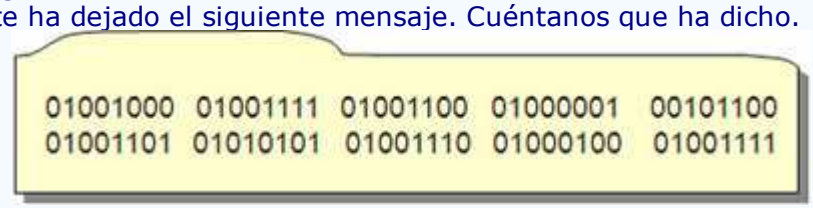
Los lenguajes programación de **bajo nivel** se basan en instrucciones orientadas a la máquina. La primera generación es el **lenguaje máquina** y la segunda generación es el **lenguaje ensamblador**.

Lenguaje máquina

Es el lenguaje que entiende el ordenador y utiliza el código binario. Fue el primero que se utilizó, pero tiene algunas desventajas que han hecho que prácticamente ya no se utilice. Aunque los programas se cargan directamente en la memoria, con lo cual la velocidad de ejecución es alta, no se puede utilizar el mismo programa en ordenadores de diferentes características. También es poco manejable porque las instrucciones son difíciles de escribir y de memorizar.

Autoevaluación

¿Serías capaz de hablar al ordenador en su propio idioma? Como ya sabes, el lenguaje máquina se basa en el código binario y para traducirlo a nuestro lenguaje, tenemos que utilizar el código ASCII.
El ordenador te ha dejado el siguiente mensaje. Cuéntanos que ha dicho.



En las siguientes páginas web puedes encontrar la correspondencia entre el código ASCII, el sistema decimal y el sistema binario:

Traductor código ASCII
Convertidor binario-decimal

Lenguaje ensamblador

Este lenguaje sustituye el código máquina utilizando instrucciones formadas por palabras alusivas a la funcionalidad que tienen en el programa. Estas instrucciones se llaman nemotécnicos.

Es más sencillo que el lenguaje máquina, pero no puede ser ejecutado directamente por el ordenador, necesitando la mediación de un programa traductor que es el llamado ensamblador. Los primeros ensambladores surgieron en la década de los 50, pero en la actualidad poseen grandes posibilidades de abstracción, lo cual los hace más fáciles de manejar. Los ensambladores avanzados permiten utilizar procedimientos de alto nivel, declaración de funciones, variedad en los tipos de datos (estructuras, registros, uniones) y procesamiento de macros.

Img 4. Estructura de un programa ensamblador

Imagen obtenida en Universidad Carlos III con licencia Creative Commons

Autoevaluación

1) Una ventaja del lenguaje máquina es:

- a) Se puede utilizar en distintos ordenadores.
- b) Difícil de escribir.
- c) Alta velocidad de ejecución.

2) El ensamblador es:

- a) Un programa traductor.
- b) Instrucciones de un lenguaje ensamblador.
- b) Difícil de escribir.

3) Las instrucciones que aparecen en la imagen anterior se llaman:

- a) Datos.
- b) Código.
- c) Nemotécnicos.

Curiosidad

Si, es otra vez el programa **"HOLA, MUNDO"**.

En este caso, está escrito en un lenguaje ensamblador para la arquitectura de procesador x86, bajo el sistema operativo DOS.

```
.model small
.stack
.data Cadena1 DB 'Hola Mundo.$'
.code

programa:
        mov ax, @data
        mov ds, ax
        mov dx, offset Cadena1
        mov ah, 9 int 21h
end programa
```

2.2 Lenguajes de alto nivel

Importante

Los lenguajes de **alto nivel** son aquellos que utilizan el lenguaje natural para realizar programas y, por tanto, necesitan otro programa para generar el código máquina ya que el ordenador no los entiende directamente. Este programa puede ser:

- Un **intérprete**.- traduce cada línea del programa siguiendo la secuencia; ejecuta el programa paso a paso hasta el final o hasta que encuentra un error, en cuyo caso, se detiene.
- Un **compilador**.- traduce el programa completo creando otro que entiende en ordenador y que se llama programa objeto; si encuentra errores, indica su posición para que sean corregidos. El programa objeto obtenido es el ejecutable.

Img 5. Lenguajes de alto nivel

Imagen de producción propia

La gran ventaja de los lenguajes de alto nivel es que consiguen distanciarse del lenguaje máquina y se aproximan al lenguaje natural.

Uno de los mayores problemas de estos lenguajes es la cantidad de ellos que existen y las nuevas versiones que aparecen continuamente.

A continuación vamos a describir algunos de los lenguajes más conocidos. En el tema 3 de esta unidad, aprenderás uno de ellos.

FORTRAN ▸ Es el primer lenguaje de programación de alto nivel y fue creado en 1955 por IBM para resolver ecuaciones algebraicas de uso científico. ▸ Está especializado en aplicaciones técnicas y científicas y se caracteriza por su potencia en los cálculos matemáticos. Sin embargo, su uso es limitado en las aplicaciones de gestión, manejo de archivos y edición de informes, aunque las últimas versiones han evolucionado también en este sentido. ▸ A lo largo de la historia, han ido apareciendo distintas versiones: FORTRAN IV, FORTRAN 77, FORTRAN 80 Y FORTRAN 90. Debido a sus características, ha sido adoptado por la comunidad científica para cálculos intensivos.	 Img 6. Logo FORTRAN Wikimedia Commons - licencia CC
COBOL ▸ Fue creado en 1960 en Estados Unidos con el fin de disponer de un lenguaje universal para aplicaciones comerciales y constituyó el origen de la llamada informática de gestión. ▸ Destaca su capacidad para manejar ficheros y tablas y en la producción de informes. Los mayores inconvenientes son la rigidez de las reglas de formatos de escritura, la extensión excesiva en sus sentencias, la inexistencia de funciones matemáticas. Se utiliza en sistemas que requieren gran capacidad de procesamiento por lotes. ▸ También ha evolucionado en sus distintas versiones hasta la actualidad en los modelos COBOL-ANSI y COBOL-ENTRERPRISE.	 "HOLA, MUNDO" en COBOL

BASIC

- Fue diseñado en 1965 por los profesores John G. Kemeny y Thomas E. Kurtzun para crear un lenguaje sencillo para utilizar en educación. La popularización de los ordenadores personales hizo que se extendiese su uso y se convirtiese en un lenguaje útil para todo tipo de aplicaciones.
- Está disponible para casi todas las plataformas y sistemas operativos. Las versiones más conocidas de los compiladores son QBASIC, Visual BASIC (de Microsoft), RealBASIC (de MAC OS) y FreeBASIC (versión libre).

```
PRINT "HOLA MUNDO"
```

"HOLA, MUNDO" en BASIC

PASCAL

- Fue creado en 1970 por el matemático suizo Niklaus Wirth, basándose en el lenguaje ALGOL (*Algorithmit Language*), y su objetivo era proporcionar un lenguaje para enseñar técnicas de programación.
- Con el tiempo ha llegado a ser un lenguaje ampliamente utilizado en todo tipo de aplicaciones y en la enseñanza de la programación estructurada. Aporta los conceptos de tipo de datos, programación estructurada y diseño descendente.
- Es el predecesor de otros lenguajes como MODULA-2, DELPHI y ADA.

C

- Fue creado en 1972 por Dennis Ritchie para conseguir un lenguaje que permitiera crear sistemas operativos. Posteriormente se ha extendido a aplicaciones técnico-científicas, bases de datos o proceso de textos, etc. Con el se creó el sistema operativo UNIX, que es dónde su utilización es óptima.
- Su evolución ha sido hacia la programación orientada a objetos, creándose el C++ y el Objective C.
- Entre sus características destaca el uso de programación estructurada, acceso a dispositivos hardware a bajo nivel y el amplio número de librerías de las que dispone, con rutinas y ficheros estandarizados por la ISO. Se dice que es un lenguaje de nivel medio puesto que conjuga las facilidades de los lenguajes de alto nivel con la potencia de los de bajo nivel.

```
# include <iostream> // Esta biblioteca permite el uso
de cout(<<) y de cin(>>)

using namespace std;

int main()
{
        cout << "Hola mundo" << endl;
        return 0;
}
```

"HOLA, MUNDO" en C

Autoevaluación

Como imaginarás, los nombres de los lenguajes de programación no surgen de la nada. Algunos de ellos son acrónimos, otros tienen nombres de sus creadores o de científicos y otros son simplemente curiosos.
Te proponemos que averigües la procedencia de los nombres de los lenguajes de programación que aparecen en este punto.

FORTRAN	COBOL	BASIC	PASCAL	C

Para saber más

Los lenguajes de programación no se crean, se utilizan y mueren. Podría decirse que se reproducen creando nuevas generaciones. Ese es el caso de PASCAL, que ha sido la base de otros dos lenguajes de programación:

MODULA-2

A finales de los años 70, Nicklaus Wirth, creador del lenguaje PASCAL, dirige el desarrollo del MODULA-2 con la intención de incluir las necesidades de la programación de sistemas del PASCAL. Este nuevo lenguaje supera las carencias del PASCAL y ha sido adoptado como herramienta para la enseñanza de la programación.

ADA

Se publicó en 1983 con el nombre de ADA en honor de la considerada primera programadora de la historia Augusta Ada Byron, condesa de Lovelace. Tiene grandes ventajas como la compilación separada y los tipos abstractos de datos, pero su mayor inconveniente es su gran extensión.

DELPHI

Es una herramienta visual basada en lenguaje PASCAL. La primera versión de DELPHI fue lanzada por Borland en 1994 y es un lenguaje capaz de generar aplicaciones de menor tamaño y mucho más rápidas que las de productos similares.

Img 6-B. Lenguaje de programación ADA

Autoevaluación

Cada lenguaje de programación tiene sus características diferenciadoras. ¿Eres capaz de identificarlos?

Indica en la siguiente lista el nombre del lenguaje al que se hace referencia: FORTRAN, COBOL, BASIC, PASCAL ó C.

Es un lenguaje destinado a aplicaciones comerciales y de gestión.	
Está disponible en todas las plataformas y sistemas operativos.	
Dispone de una librería de rutinas estandarizada.	
Está especializado en aplicaciones técnicas y científicas.	
Se creó para la enseñanza de técnicas de programación.	

Comprobar

2.3 Lenguajes de 3ª generación

Este segundo grupo de lenguajes de programación de alto nivel constituyen la llamada 3ª generación. Son lenguajes claramente orientados a internet.

PERL

- Fue diseñado por Larry Wall en 1987 y destaca por no tener ninguna de las limitaciones de los otros lenguajes de script.
- Es un lenguaje especializado en el procesamiento de textos, particularmente extraer y validar las respuestas a cuestionarios incluidos en páginas Web.

```
# !/usr/bin/perl
use 5.010;
say '¡Hola mundo!';
```

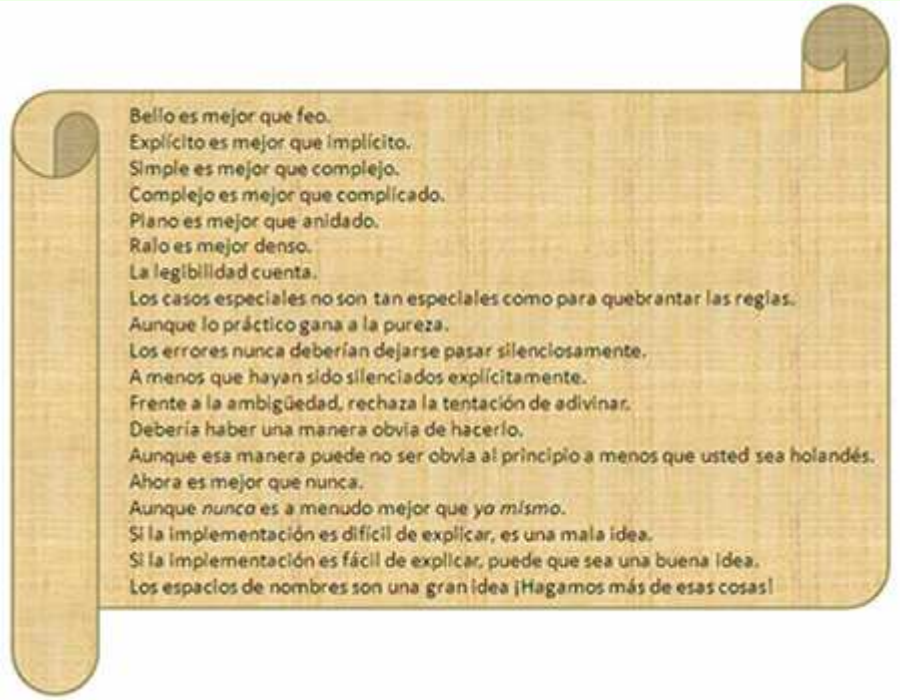
"HOLA, MUNDO" en Perl

PYTHON

- Es un lenguaje de programación interpretado creado por Guido van Rossum en el año 1991. Python es el lenguaje opositor a Perl ya que se considera mucho más limpio y elegante para programar.
- En la actualidad Python se desarrolla como un proyecto de código abierto.

Curiosidad

El desarrollador de Python, Tim Peters, describió la filosofía del lenguaje en una serie de principios de legibilidad y transparencia, contrarios al código opaco u ofuscado. Este código se dice que es *"pythonico"* y constituye *El Zen de Python*. ¿Curioso verdad?



Img 7. El Zen de Python

Imagen de producción propia

JAVA

- Fue presentado por Sun Microsystems en 1995. James Gossling y su equipo desarrollaron un nuevo lenguaje de programación capaz de adecuarse a cualquier entorno de ejecución (portable) y basado en la simplicidad.
- JAVA es un lenguaje de programación orientado a objetos, independiente de la plataforma en la que se ejecute y preparado para trabajar en internet.

```
// Hola.java
public class Hola
{
        public static void main(String[] args)throws IOException {
        System.out.println("¡Hola, mundo!");
        }
}
```

"HOLA, MUNDO" en Java

Para saber más

Applets de java

Seguro que has oído estas palabras muchas veces, pero ¿qué es un applet?

Un applet es un componente de una aplicación que se ejecuta en un navegador web. Un applet Java es un applet escrito en el lenguaje de programación Java que se ejecutan utilizando la Java Virtual Machine (JVM).

Estos son sólo algunos ejemplos y en los enlaces siguientes encontrarás muchos más.

recopilación de applets de java

applets de física

applets de matemáticas

[redacted]

RUBY

- Fue creado por Yukihiro Matsumoto en el año 1993 en Japón. Es un lenguaje de programación basado en el paradigma de la orientación a objetos.
- Es rápido y sencillo ya las variables no necesitan se declaradas, tiene una sintaxis clara y simple y la gestión de memoria se realiza automáticamente.

```
#!/usr/bin/ruby
print "hola mundo"
```

"HOLA, MUNDO" en Ruby

Autoevaluación

Repasa las características de los últimos lenguajes de programación e indica a cuál corresponde cada una de ellas.

Es un lenguaje capaz de adecuarse a cualquier entorno de ejecución (portable) y basado en la simplicidad.	
Es un lenguaje especializado en el procesamiento de textos, particularmente incluidos en páginas Web.	
En la actualidad se desarrolla como un proyecto de código abierto.	
Es rápido y sencillo ya las variables no necesitan se declaradas, tiene una sintaxis clara y simple y la gestión de memoria se realiza automáticamente.	

Comprobar

Autoevaluación

Ahora que conoces los nombres de lenguajes de programación de alto nivel, te proponemos que busques los siguientes en la sopa de letras.

ADA
BASIC
COBOL
DELPHI
FORTRAN
JAVA
PASCAL
PERL
PYTHON
RUBY

w	r	e	p	i	n	u	p	p
f	u	o	y	h	y	i	a	x
o	b	h	t	p	s	k	s	d
r	y	y	h	l	e	r	c	q
t	f	x	o	e	c	r	a	b
r	j	m	n	d	o	s	l	a
a	d	a	m	v	b	q	y	s
n	a	g	v	w	o	x	l	i
h	u	d	j	a	l	t	d	c

2.4 Lenguajes de 4ª generación

Los **lenguajes de 4ª generación** o **4GL** especifican qué resultados se quieren obtener y no cómo deben obtenerse. No es necesario definir los pasos a seguir en un programa para realizar una tarea determinada, sino una serie de parámetros que serán utilizados para generar un programa.

Quizás el lenguaje más conocido de los 4GL es el SQL. Este lenguaje lo aprenderás en profundidad en la próxima unidad.

SQL

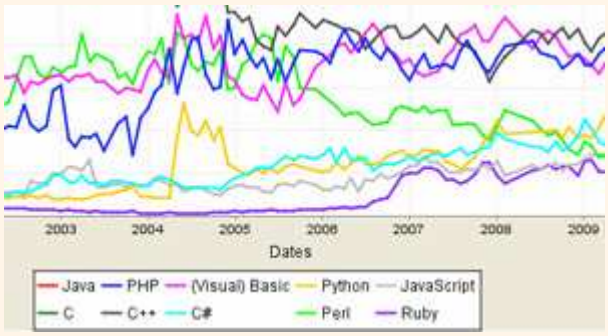
- El Lenguaje de Consulta Estructurado SQL (Structured Query Language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite realizar operaciones como consultas o modificaciones en éstas. Su predecesor es el SEQUEL de IBM, aunque fue Oracle quien lo desarrolló. En 1986 fue publicado y el año siguiente confirmado por la ISO.
- Una de sus características es el manejo del álgebra y el cálculo relacional. Permite una alta productividad en codificación ya que se orienta al manejo de conjuntos de registros, y no a registros individuales. Una sola sentencia puede equivaler a uno o más programas en un lenguaje de bajo nivel.
- Los sistemas más conocidos que utilizan SQL son DB2, MySQL, ORACLE y SQL Server.

Para saber más

Los lenguajes de programación están en constante evolución y su uso varía en función de la demanda de los programadores. Si te interesa saber cuáles son los lenguajes más utilizados, en el siguiente enlace encontrarás gráficos actualizados mensualmente.

Clasificación lenguajes de programación

Estos gráficos se publican en la revista *Tiobe Software* y se realizan basándose es aspectos tales como el número de ingenieros cualificados en todo el mundo que utiliza cada lenguaje, los cursos que se ofertan y los proveedores.



Img 8. Gráfico uso de lenguajes de programación

Imagen de Tiobe Software