Tema 1: Programas informáticos y lenguajes de programación

¿Qué aprenderás?

- Identificar los conceptos más importantes de los programas informáticos.
- Diferenciar los elementos de un programa.
- Reconocer y clasificar los diferentes tipos de lenguajes de programación.
- El proceso de obtención de código ejecutable.
- Determinar las fases del desarrollo de software.
- Localizarás personas involucradas en el desarrollo de software.

¿Sabías que…?

- El sistema sólo es capaz de entender código escrito en código máquina (1s y 0s).
- El lenguaje ensamblador es el primer lenguaje de programación qué ha hecho servir códigos mnemotécnicos.

1.1. Conceptos de los programas informáticos

1.1.1. Definición de programa informático

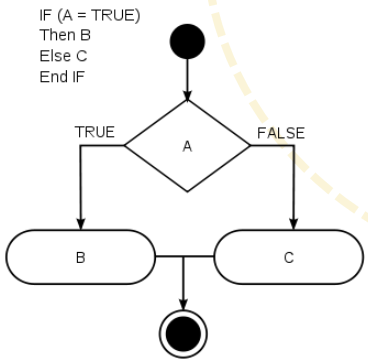
Un programa informático es un conjunto de instrucciones o pasos, escritos en un lenguaje de programación que se ejecutan de manera secuencial. Tienen el objetivo de realizar una o varias tareas en un sistema.

- **Es algo secuencial...**

Estas instrucciones en realidad se dividen subinstrucciones o microinstrucciones que se ejecutarán una a una y de forma secuencial en cada ciclo del procesador (microinstrucciones).

Nosotros (los programadores) cuando desarrollamos un software, lo estamos desarrollando en un programa de un lenguaje de programación. Probablemente conozcas el lenguaje Java, que tiene cierta semejanza al inglés.

Por ejemplo, una instrucción que para nosotros los programadores informáticos puede resultar una instrucción muy sencilla IF-THEN-ELSE se divide en muchas microinstrucciones.



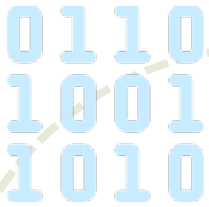
- **¿Qué datos procesa un programa informático?**

Depende del tipo de programa. Vemos algunos ejemplos.

- **Un navegador web**: procesa órdenes de un usuario y órdenes que recibe del servidor.
- **Un videojuego**: procesa la información referente a la ubicación de enemigos y jugadores así como las físicas del juego, los impactos, la puntuación, etcétera,
- **Un programa ofimático**: procesa datos de texto datos numéricos, imágenes.

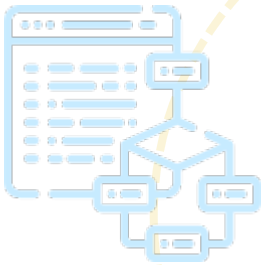
1.2. Conceptos de los lenguajes de programación

Un sistema informático sólo es capaz de entender código escrito en código máquina (1s y 0s). Así que programar directamente en código máquina es una muy complicada.



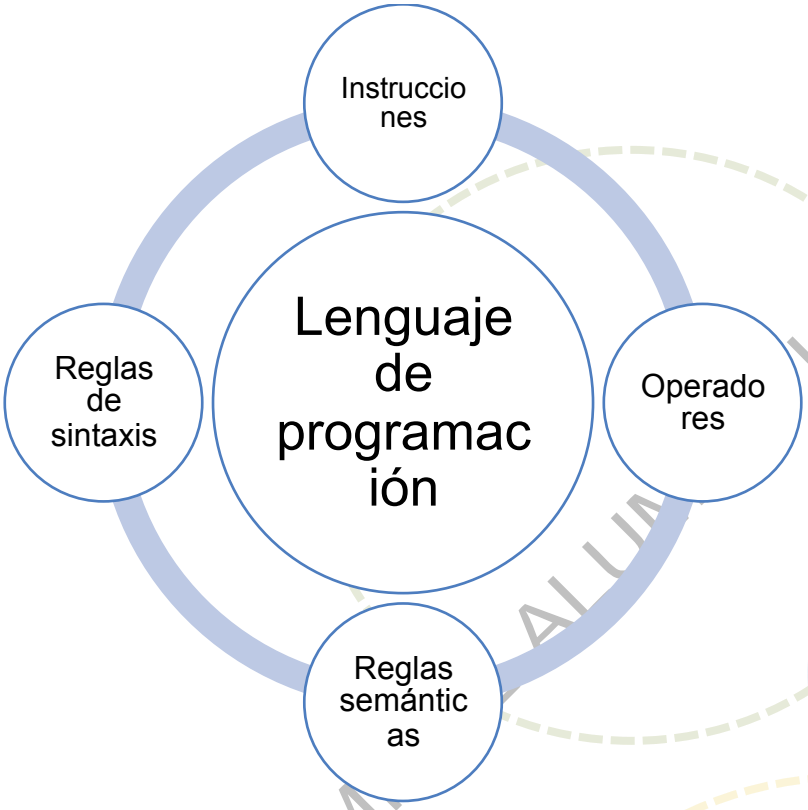
Si queremos que nuestro código pueda interactuar con el procesador nos vemos obligados a hacer servir un lenguaje de programación. Más que nada porque el sistema y nuestro procesador tan sólo es capaz de interpretar unas instrucciones a muy bajo nivel, es decir con ceros y unos.

Debido a esta dificultad, aparecen los lenguajes de programación. El uso de un lenguaje de programación tiene el objetivo de facilitar la tarea a los programadores permitiendo escribir programas utilizando un mayor nivel de abstracción en el código.



1.2.1. Definición de lenguaje de programación

Conjunto de **instrucciones** + **operadores** + **reglas de sintaxis y semánticas** que se ponen a disposición de los programadores para que se comuniquen con el hardware y el software.

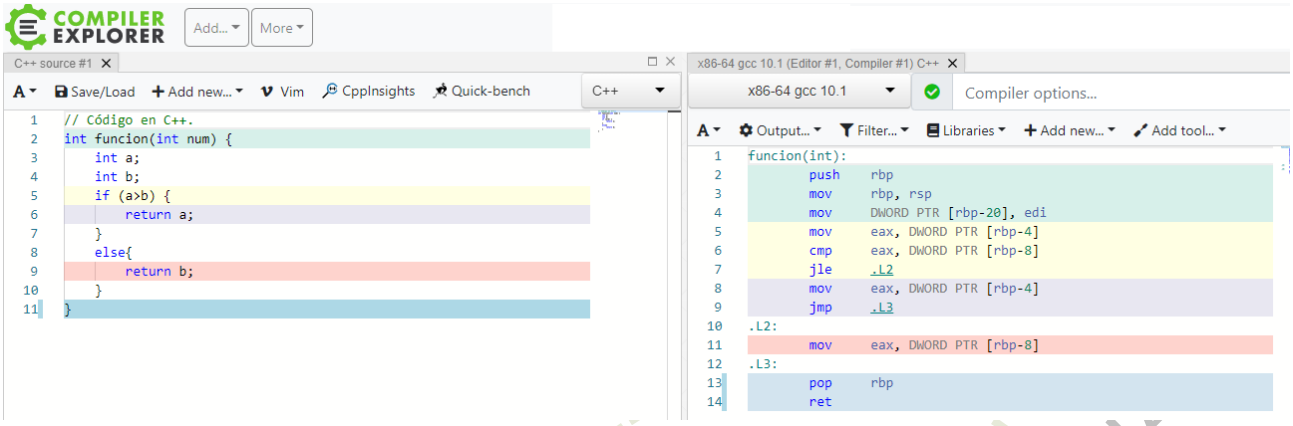


Estas instrucciones se ponen a disposición de los programadores para que redacten ellos sus códigos y se pueda hacer un código entendible por los humanos.

Más adelante veremos que existen diferentes niveles de lenguajes de programación.

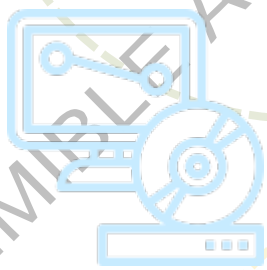
La tendencia actual usar **lenguajes de alto nivel**, que son **más cercanos al lenguaje humano**. También es posible programar a medio y bajo nivel pero se convierte en una tarea muy complicada.

Podemos ver en la siguiente imagen el contraste entre un lenguaje de programación de alto nivel como C++ y otro de más bajo nivel como es assembler.



Con la herramienta online compiler explorer puedes traducir de los lenguajes de alto nivel más comunes a lenguaje ensamblador (assembler)

Hay programadores que todavía desarrollan en lenguaje assembler. Normalmente lo usan para desarrollar los drivers de ciertos periféricos.



Estos lenguajes de muy bajo nivel ocupan muy poco y son óptimos para estos pequeños programas que comunican el dispositivo con nuestro ordenador.

1.2.2. Características de los lenguajes de programación más usados

La programación ha ido evolucionando a lo largo de los años, con cada evolución se han ido creando lenguajes de programación cada vez más eficientes y fáciles de hacer servir.

Seguro que alguna vez has visto algún lenguaje de programación totalmente visual que permite a personas sin conocimientos técnicos de informática crear pequeños programas.

Los diferentes tipos de lenguajes son:

1.2.2.1. Lenguajes de primera generación. El lenguaje máquina

Es el lenguaje que entiende el ordenador directamente, estamos hablando a nivel de procesador. Sólo hay uno y es el código máquina.

Las instrucciones se expresan en código binario (0 y 1). Eso sí, el código **máquina hace posible que el programador pueda utilizar todos los recursos del hardware permitiendo así obtener programas muy eficientes**.

Sólo se hace servir en procesadores muy concretos o funcionalidades muy específicas.

1.2.2.2. **Lenguajes de segunda generación o lenguaje ensamblador**

Son los que permiten escribir programas muy optimizados que permiten aprovechar al máximo el hardware. También permiten al programador especificar exactamente qué instrucciones quiere que se ejecuten.

Cómo inconveniente estos lenguajes de programación dependen directamente del hardware en el que se ejecutan y no se pueden trasladar a otro de forma fácil. También requiere conocer a fondo la arquitectura del sistema y del procesador. Lo más importante son muy difíciles de codificar y mantener.

Assembler es el primer lenguaje de programación qué ha hecho servir códigos memotécnicos.
Son instrucciones muy básicas. Este tipo de lenguaje depende directamente del procesador que utiliza la máquina. Este tipo de lenguaje se utiliza para programar controladores (drivers) o aplicaciones que requieran un uso muy eficiente de la velocidad y la memoria.

```
; Ejemplo sencillo tipo Hello World
; Escribe un string en la salida

        JMP start
hello: DB "¡Bienvenidos a Entornos de desarrollo!" ; Variable
       DB 0     ; String terminator

start:
        MOV C, hello    ; Point to var
        MOV D, 232      ; Point to output
        CALL print
        HLT             ; Stop execution

print:                  ; print(C:*from, D:*to)
        PUSH A
        PUSH B
        MOV B, 0
.loop:
        MOV A, [C]      ; Get char from var
        MOV [D], A      ; Write to output
        INC C
        INC D
        CMP B, [C]      ; Check if end
        JNZ .loop       ; jump if not

        POP B
        POP A
        RET
```

Ejemplo de lenguaje ensamblador. Puedes hacer la prueba con esta herramienta online

https://schweigi.github.io/assembler-simulator/

1.2.2.3. **Lenguajes de tercera generación. Los lenguajes de alto nivel**

Son los lenguajes que hacen servir **palabras y frases relativamente fáciles de entender**. Gracias a esto podemos expresar flujos de control de una forma bastante intuitiva. Estos lenguajes de alto nivel se hacen servir para desarrollar grandes aplicaciones. Son lenguajes de programación **independientes de la máquina en la que se van a ejecutar**.

```
1  # Ejemplo en Python
2  x = 1
3  if x == 1:
4      print("x vale 1.")
```

Ejemplo de unas líneas en lenguaje Python

Los lenguajes de alto nivel son **fáciles de aprender** porque están formados por elementos del lenguaje natural, normalmente palabras en inglés.

En resumen, el código de este tipo de programas es sencillo y comprensible y por ese motivo que es más sencillo ejecutarlos en diferentes máquinas de diferentes arquitecturas o sistemas operativos (esto va a depender del compilador). En contrapartida ejecutar un programa escrito en alto nivel será más lento que haberlo hecho en un lenguaje de bajo nivel.

1.2.2.4. **Lenguajes de cuarta generación o de propósito específico**

Los lenguajes de propósito específico permiten desarrollar **aplicaciones sofisticadas** en poco tiempo. Estos lenguajes permiten muchas acciones que antes se tenían que hacer manualmente como por ejemplo realizar consultas a una base de datos con una sola instrucción. Por ejemplo una query SELECT en una línea.

SQL Statement:

```
SELECT NombreCliente,Ciudad FROM Clientes;
```

Ejemplo de una sentencia Select en SQL. Puedes hacer pruebas con un simulador online como el de w3schools

Estos lenguajes de programación están orientados básicamente al **manejo de base de datos**.

1.2.2.5. **Lenguajes de quinta generación**

Son lenguajes específicos para tratar problemas relacionados con la **inteligencia artificial.**

```
% La sintaxis es factorial(N, F) -> Factorial de N es F (el resultado se guarda en F)
factorial(0, 1).
factorial(1, 1).
factorial(N, F) :- N>0, N1 is N - 1, factorial(N1, F1), F is N * F1.

%el factorial se llama recursivamente dejando el resultado en F
```

Ejemplo del cálculo del factorial de un número en lenguaje Prolog. Ejemplo de la wikipedia

1.2.3. El nivel de abstracción en los lenguajes de programación

El nivel de abstracción de un lenguaje **implica lo alejado que está del código máquina**. Cuando más parecido sea a nuestro lenguaje y menos al código máquina mayor será nivel del lenguaje.

Por ejemplo el lenguaje Java es alto nivel, el código assembler es de medio nivel y el código máquina (ceros y unos) es de bajo nivel.

- **Bajo nivel -** Sólo hay uno el código máquina ceros y unos**.**
- **Medio nivel -** El lenguaje ensamblador que hace servir instrucciones sencillas para trabajar con datos simples y posiciones de memoria.
- **Alto nivel -** Todos los demás lenguajes de programación son los que son más cercanos a nuestro lenguaje

1.2.4. Los lenguajes de programación según la forma de ejecución

- **Compilados**

Son los lenguajes que deben ser compilados antes de poder ejecutarse.

Normalmente, los programadores codifican el programa (esto es el código fuente), este código fuente necesitaremos que sea validado y compilado con tal de que llegue a ser ejecutable. La compilación es el proceso que consigue que el lenguaje de programación baje de nivel hasta el código máquina y sea capaz de ejecutarse.

- **Interpretados**

Estos lenguajes **se ejecutan línea a línea** es decir interpreta una línea y realiza la acción que está indica una vez realizada pasa a la siguiente línea y así sucesivamente.

A diferencia de los lenguajes de programación compilados **no necesitamos compilar el programa completo para ejecutarlo**.

Estos lenguajes ejecutan las instrucciones directamente sin generar un código objeto. El código interpretado no lo ejecuta directamente el sistema operativo, sino que lo hace un intérprete. El sistema tiene su propio intérprete.

> *El **intérprete** es un programa que traduce el código de alto nivel a lenguaje máquina, a diferencia del compilador el intérprete lo hace en tiempo real. es decir no hace un proceso traducción de todo el código fuente antes de ejecutar sino que va traduciendo y ejecutando cada instrucción una tras otra.*

Por ejemplo si las primeras 10 primeras líneas son correctas el interprete las irá ejecutando correctamente si la línea número 11 tiene algún error en ese punto se producirá el error.

- **Virtuales**

Son parecidos a los lenguajes compilados, a partir del código fuente, **se llega a compilar pero no se genera un ejecutable propiamente para la máquina en la que se está compilando**.

En los lenguajes compilados cuando compilas obtienes un ejecutable propio para tu hardware. En cambio, **los lenguajes de programación virtuales usan una máquina virtual**.

En definitiva esta máquina virtual es capaz de entender este código intermedio bytecode y se encarga de hacerlo ejecutar en la máquina física en la que está instalada la máquina virtual.

1.2.5. Los lenguajes de programación según el paradigma de programación

El paradigma de programación de un lenguaje de programación se basa en:

- El **método** para llevar a cabo los cálculos en el proceso.
- La **forma** en la que deben estructurarse las tareas que debe realizar el programa.

Los paradigmas se diferencian unos de otros en la forma de abstraer los elementos del lenguaje de programación así como de los pasos que se deben seguir para llegar a la solución del problema.

- **Lenguajes imperativos o estructurados**

Se basan en **sentencias imperativas**. Es decir que realizan una determinada operación una tras otra. Estas operaciones van modificando los datos de la memoria.

Para estos lenguajes de programación imperativos se hace servir la técnica de la programación estructurada, es decir de un programa grande y complejo se divide y se representa con secuencias, selecciones, iteraciones, etc.

Para este tipo de programas normalmente se trabaja dividiendo el programa en módulos y así conseguir porciones más pequeñas de código con tareas específicas, estos módulos también se dividen y se crean funciones más pequeñas y reutilizables.

- **Orientado a objetos**

Lenguajes que **intentan abstraer conceptos de la vida real y representarlos con objetos**.

En un programa orientado a objetos la abstracción no son los procedimientos ni las funciones sino los objetos. Un objeto es una combinación de datos y métodos diseñados para interactuar entre objetos.

- **Funcional**

Son lenguajes **basados en modelos matemáticos**. Funcionan teniendo en cuenta en que el resultado de un cálculo es la entrada del siguiente, siempre de forma sucesiva hasta que se produce un resultado.

Gracias a estos lenguajes podemos tener código reutilizable y permitir las interacciones entre las diferentes funciones.

Normalmente estos lenguajes se usan en ámbitos de investigación científica y aplicaciones matemáticas.

- **Lógico**

Son lenguajes en los basados en modelos matemáticos y que tiene la finalidad de acabar respondiendo preguntas planteadas al sistema para resolver problemas.

Mejor un ejemplo para simplificar esta definición…
Un programa necesita una base de conocimientos formadas por hechos, los hechos son los que representan la información.

Por ejemplo: unos hechos podrían ser: Cristian tiene una moto, Cristian tiene carné de conducir moto.

A demás se necesitan reglas lógicas que permitan deducir las consecuencias de combinar los hechos.

Una regla lógica podría ser que sólo es posible conducir una moto si el conductor tiene moto y carné de conducir.

Con esta información podríamos diseñar un programa para verificar conductores. A partir de estos hechos se define la lógica y si se cumple retornar a qué es cierto. Estos lenguajes normalmente **se hacen servir para investigación**.

Recursos y enlaces

- Compiler Explorer (herramienta online que traduce de lenguaje de alto nivel a assembler)

- Assembler Simulator (herramienta online que permite 'ejecutar' un código en assembler y ver como se modifican los registros de memoria en tiempo real)

Conceptos clave

- Los lenguajes de programación nos facilitan la tarea de programar acercándose al lenguaje humano.
- Un lenguaje de programación está formado por un conjunto de instrucciones más una serie de operadores y unas reglas de sintaxis y semánticas.
- Existen diferentes tipos y generaciones de los lenguajes de programación.
- El nivel de abstracción de los lenguajes de programación los clasifica según lo alejado que está del código máquina.
- Se pueden clasificar los lenguajes de programación según su forma de ejecución.
- El paradigma de los lenguajes de programación indica el método de programación y la forma de programar

Test de autoevaluación

1. Un programa informático es…
   a) una única instrucción escrita en un lenguaje de programación que se ejecuta de manera secuencial.
   b) es un conjunto de instrucciones o pasos, escritos en un lenguaje de programación que se ejecutan de manera paralela.
   c) es un conjunto de instrucciones o pasos, escritos en un lenguaje de programación que se ejecutan de manera secuencial.
   d) una única instrucción escrita en un lenguaje de programación que se ejecuta de manera paralela.

2. Un lenguaje de programación está formado por…
   a) Conjunto de instrucciones + reglas de sintaxis y semánticas.
   b) Conjunto de instrucciones + operadores.
   c) Conjunto de instrucciones + operadores + reglas de sintaxis y semánticas.
   d) Conjunto de instrucciones + operadores + reglas de sintaxis y semánticas + compilador.

3. El nivel de abstracción de un lenguaje implica…
   a) lo alejado que está del código máquina. Cuando más parecido sea a nuestro lenguaje y menos al código máquina mayor será nivel del lenguaje.
   b) lo alejado que está del código máquina. Cuando más parecido sea a nuestro lenguaje y menos al código máquina menos será nivel del lenguaje.

Ponlo en práctica

Actividad 1

1. Clasificar lenguajes de programación.

1.1. Rellenar la siguiente tabla con diferentes lenguajes de programación.

Tipo de lenguaje	Ejemplos de lenguajes
De primera generación	
De segunda generación	
De tercera generación*	
De cuarta generación*	
De quinta generación	

SOLUCIONARIOS

Test de autoevaluación

1. Un programa informático es…

   e) una única instrucción escrita en un lenguaje de programación que se ejecuta de manera secuencial.

   f) es un conjunto de instrucciones o pasos, escritos en un lenguaje de programación que se ejecutan de manera paralela.

   g) es un conjunto de instrucciones o pasos, escritos en un lenguaje de programación que se ejecutan de manera secuencial.

   h) una única instrucción escrita en un lenguaje de programación que se ejecuta de manera paralela.

2. Un lenguaje de programación está formado por…

   e) Conjunto de instrucciones + reglas de sintaxis y semánticas.

   f) Conjunto de instrucciones + operadores.

   g) Conjunto de instrucciones + operadores + reglas de sintaxis y semánticas.

   h) Conjunto de instrucciones + operadores + reglas de sintaxis y semánticas + compilador.

3. El nivel de abstracción de un lenguaje implica…

   c) lo alejado que está del código máquina. Cuando más parecido sea a nuestro lenguaje y menos al código máquina mayor será nivel del lenguaje.

   d) lo alejado que está del código máquina. Cuando más parecido sea a nuestro lenguaje y menos al código máquina menos será nivel del lenguaje.

Ponlo en práctica

Actividad 1

1. Clasificar lenguajes de programación.

1.1. Rellenar la siguiente tabla con diferentes lenguajes de programación.

Tipo de lenguaje	Ejemplos de lenguajes
De primera generación	Lenguaje maquina
De segunda generación	Lenguaje ensamblador
De tercera generación*	C, Pascal
De cuarta generación*	PHP, .NET
De quinta generación	Lisp, Prolog