TEMA1

EL SOFTWARE

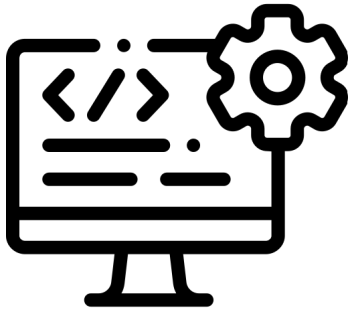
Entornos de Desarrollo

1 DAM/DAW

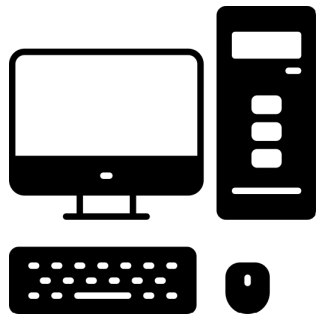
Índice

1. Introducción

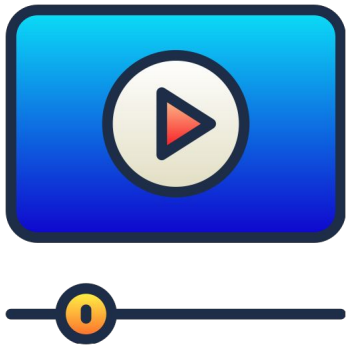
¿ES IMPORTANTE LA TECNOLOGÍA QUE VAYAMOS A USAR?



Software: Conjunto de programas o aplicaciones, instrucciones y reglas informáticas que hacen posible el funcionamiento del equipo.



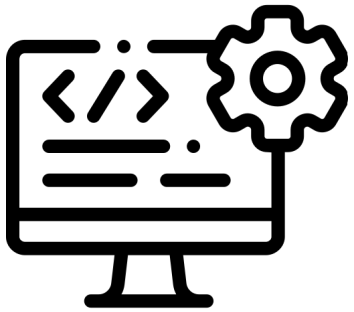
Hardware: conjunto de componentes físicos de los que está hecho el equipo.



https://www.youtube.com/watch?v=3F-kuNGlNco

1. Introducción

Tipos de Software informático



Software de sistema	Software de aplicación	Software de soporte
Diseñado para facilitar el funcionamiento y mantenimiento de un sistema informático y sus programas asociados. Serían software de sistema, por ejemplo, los sistemas operativos, las utilidades del sistema o los drivers.	Diseñado para ayudar al usuario a realizar tareas concretas o resolver determinados tipos de problemas. Se trata de las aplicaciones en sí, y contemplarían software como los procesadores de texto, las hojas de cálculo, las aplicaciones de diseño, etcétera.	Dedicado al desarrollo y mantenimiento de otro software, como los compiladores, los intérpretes, los editores, etcétera.

2. Programa informático

Programa informático: receta detallada de cómo resolver un problema determinado.

Es un conjunto ordenado de instrucciones, escritas en un lenguaje de programación, para realizar una tarea en particular dentro de un ordenador, es decir, una secuencia de órdenes que le indica al ordenador qué hacer.

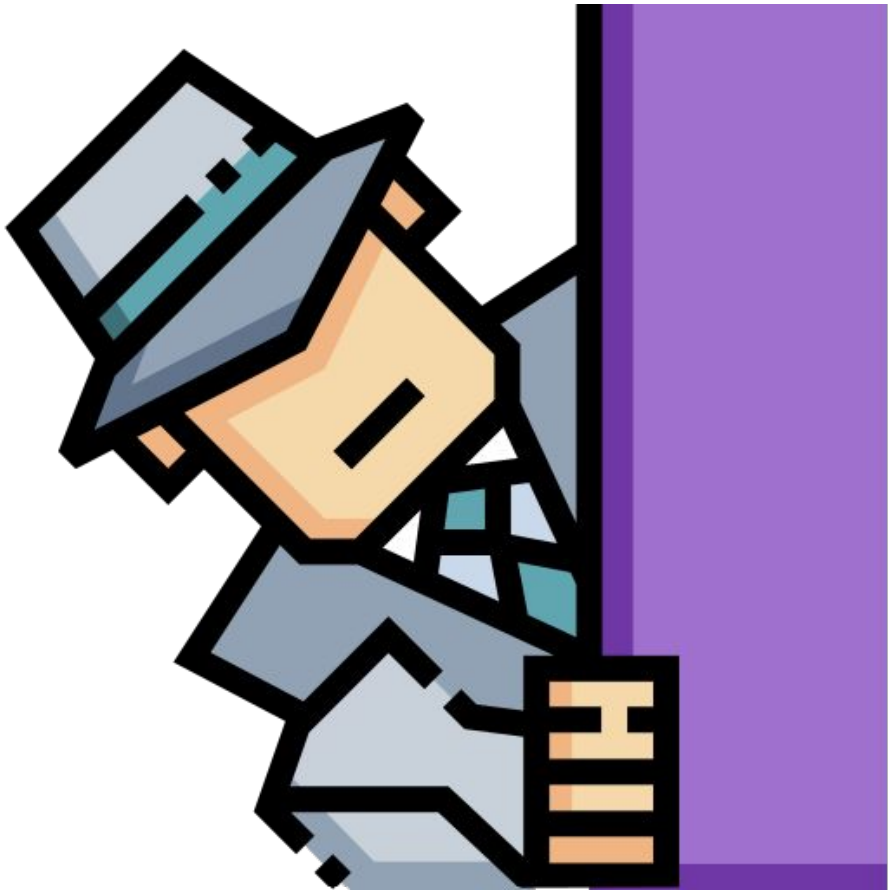
Se caracteriza por  Lógico, intangible, funcional, preciso, ejecutable y secuencial.

- Para poder ejecutar un programa informático es necesario obtener un código que pueda ser comprendido por el procesador de la máquina en la que se ejecuta.
- Cuando el programa informático va a ser ejecutado, el sistema operativo debe realizar la carga del programa, es decir, de sus datos e instrucciones en memoria principal.
- Se basa en la **Arquitectura de Von Neumann**.
- Ejemplo: Procesador de texto.

Arquitectura de Von Neumann

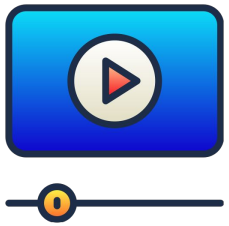
https://hardzone.es/tutoriales/rendimiento/von-neumann-limitaciones/

DE INTERÉS: INVESTIGA



3. Lenguajes de programación

Lenguaje de programación: es un conjunto de instrucciones, operadores y reglas de sintaxis y semánticas, que se ponen a disposición del programador para que este pueda comunicarse con los dispositivos de hardware y software existentes.



https://www.youtube.com/watch?v=pWw4UtQhdek&ab_channel=GCFAprendeLibre

Objetivo: es facilitar la tarea a los programadores permitiendo escribir programas utilizando un mayor nivel de abstracción del código.

- Nivel de abstracción = ocultar los detalles complejos sin que el programador se preocupe de cómo se interpreta internamente esa secuencia.

3. Lenguajes de programación

A	S	D	J	P	F	R	A	C
P	N	P	H	R	R	H	I	I
Y	O	P	B	R	S	R	T	S
T	N	I	U	T	T	O	I	D
H	F	F	S	P	F	W	F	J
O	E	L	I	C	P	R	A	A
N	T	R	I	U	H	R	Z	V
R	S	A	S	N	H	L	M	A
J	A	N	G	U	L	A	R	A

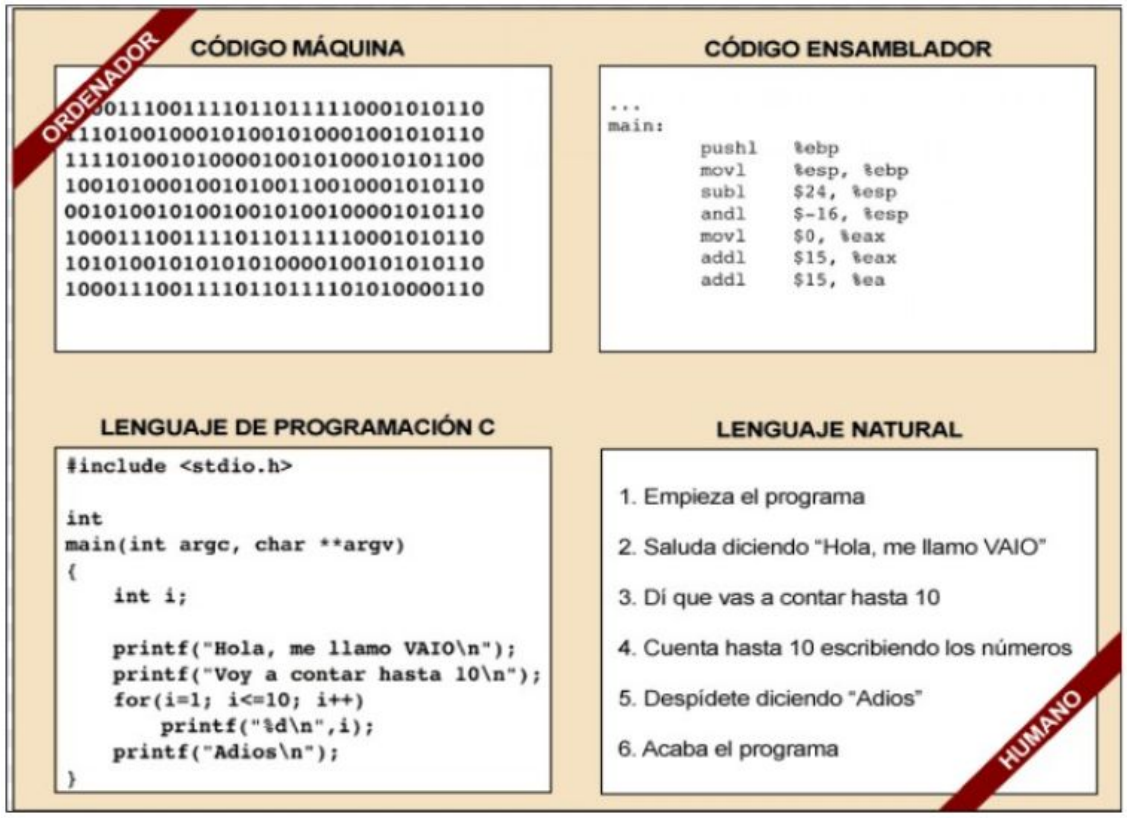
JAVA
JSCRIPT
C

PHP
PYTHON
ANGULAR

3. Lenguajes de programación

- Según el nivel de abstracción

Implica cuánto de alejado está el código máquina. Cuánto más parecido sea a nuestro lenguaje y menos al código máquina, de mayor nivel será el lenguaje.



1) **Bajo nivel o lenguaje máquina**: solo hay un lenguaje de programación = el código máquina (0 y 1).

1) **Medio nivel o lenguaje ensamblador**: tienen definidas una serie de instrucciones sencillas para trabajar con datos simples y posiciones de memoria. Es propio de cada procesador. Código ensamblador

1) **Alto nivel**: la mayoría de los lenguajes que se utilizan hoy día para programar aplicaciones son de alto nivel. Son muy cercanos a nuestro propio lenguaje y permiten la realización de patrones de diseño complejos.

3. Lenguajes de programación

- Según la forma de ejecución

1) **Compilados**: necesitan un programa traductor (compilador) para convertir el código fuente en código máquina. Se ejecutan de forma más rápida que los interpretados o los virtuales.

1) **Interpretado**: no se genera código objeto (*conjunto de instrucciones y datos que el ordenador entiende directamente).* Es un programa que tiene que estar cargado en memoria y se encarga de leer cada una de las instrucciones, interpretarlas y ejecutarlas. Se van traduciendo conforme se ejecuta.

1) **Virtual o intermedio**: el código que se genera tras la compilación es un código intermedio o bytecode que se ejecuta en una máquina virtual.

3. Lenguajes de programación

▪ Según el paradigma de programación

- Indica el método de realizar los cómputos y la manera en que se deben estructurar y organizar las tareas que debe llevar a cabo un programa.
- Es un modelo de programación que representa un estilo o forma de programar o construir programas para realizar ciertas tareas o actividades.

1) **Imperativo**: : describe la programación como una secuencia de instrucciones que cambian el estado del programa, indicando cómo realizar una tarea.

1) **Declarativo**: específica o declara un conjunto de premisas y condiciones para indicar qué es lo que hay que hacer y no necesariamente cómo hay que hacerlo.

1) **Procedimental**: el programa se divide en partes más pequeñas, llamadas funciones y procedimientos, que pueden comunicarse entre sí. Permite reutilizar código ya programado y solventa el problema de la programación spaghetti.

4) Orientado a objetos: encapsula el estado y las operaciones en objetos, creando una estructura de clases y objetos que emula un modelo del mundo real, donde los objetos realizan acciones e interactúan con otros objetos.

5) Funcional: evalúa el problema realizando funciones de manera recursiva, evita declarar datos haciendo hincapié en la composición de las funciones y en las interacciones entre ellas.

6) Lógico: define un conjunto de reglas lógicas para ser interpretadas mediante inferencias lógicas. Permite responder preguntas planteadas al sistema para resolver problemas

Tarea 1

1. Busca dos ejemplos de cada uno de los lenguajes de programación según las clasificaciones vistas.

- ✔ Según el nivel de abstracción.
- ✔ Según la forma de ejecución.
- ✔ Según el paradigma de programación.

Tarea 2

1. ¿Qué es un framework? Indica un ejemplo.
2. ¿Cuál es el lenguaje de programación más utilizado en la actualidad?

Tarea 3

Verdadero/Falso

1. El lenguaje máquina se compila.
2. Un programa en un lenguaje virtual es más rápido que en uno compilado.
3. Un framework permite crear aplicaciones más robustas.
4. Actualmente, los lenguajes más usados son interpretados.

Tarea 3

Verdadero/Falso

1. El lenguaje máquina se compila. FALSO
2. Un programa en un lenguaje virtual es más rápido que en uno compilado. FALSO
3. Un framework permite crear aplicaciones más robustas. VERDADERO
4. Actualmente, los lenguajes más usados son interpretados. FALSO

Tarea 4

Verdadero/Falso

1. Un programa es una combinación de instrucciones y definiciones de datos que permiten al hardware del ordenador realizar funciones computacionales o de control. VERDADERO
2. El software engloba programas, procedimientos, documentación, datos y equipos informáticos. FALSO

4. Fases del proceso de compilación

- El código fuente pasa por diferentes fases hasta generar el código objeto y posteriormente el código ejecutable.

Los lenguajes compilados son los más usados.

TIPOS DE CÓDIGO EN LA COMPILACIÓN DE UN PROGRAMA

https://youtu.be/9Tn-aauonq4?feature=shared

Código fuente

Conjunto de instrucciones escritas en un lenguaje de programación determinado. Es decir, es el código en el que nosotros escribimos nuestro programa.

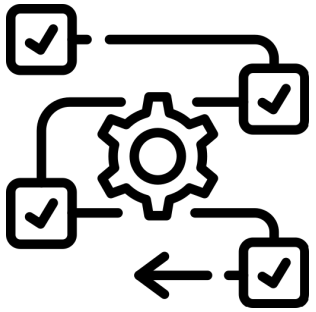
Código objeto

Es el código resultante de compilar el código fuente. Si se trata de un lenguaje compilado, el código objeto será código máquina, mientras que si se trata de un lenguaje virtual, será código bytecode.

Código ejecutable

Es el resultado de enlazar el código objeto con las librerías. Es nuestro programa ejecutable, programa que se ejecutará directamente en nuestro sistema o sobre una máquina virtual.

4. Fases del proceso de compilación



1. Análisis léxico:
2. Análisis sintáctico
3. Análisis semántico
4. Generación de código intermedio
5. Optimización c. intermedio
6. Generación de código final

4. Fases del proceso de compilación

1. **Análisis léxico**. Esta fase está centrada en comprobar que los elementos que se utilizan en la entrada de nuestro archivo pertenecen al lenguaje de programación que estamos utilizando.

2. **Análisis sintáctico.** Permite determinar si los elementos que provienen del analizador léxico vienen en el orden correcto.

3. **Análisis semántico.** Se centra en determinar si las sentencias escritas por el programador tienen sentido.

4. **Generación de código intermedio**. Una vez que finalizan todas las fases de análisis se supone que el código del programador no tienen ningún error previo, por lo tanto se puede generar una representación intermedia. Nótese que esta fase es opcional. La representación intermedia es independiente del procesador en el que se va a ejecutar el programa.

5. **Optimización de código intermedio**. Al igual que en la fase anterior se trata de una fase opcional cuyo objetivo es intentar mejorar el código generado para tener un mejor rendimiento.

6. **Generación de código final.** En esta fase somos capaces de generar código objeto que dependerá del conjunto de instrucciones de la CPU utilizada.

Compilación

¿ALGUNA DUDA?

GRACIAS