



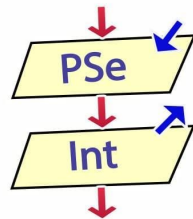
Agencia de
Aprendizaje
a lo largo
de la vida

CODO A CODO INICIAL

Clase 1

Algoritmia 1

Algoritmia y Diagramas de Flujo



Les damos la bienvenida

Vamos a comenzar a grabar la clase

Clase 00

Presentación del curso

- Sobre Codo a Codo.
- Objetivo del curso y lineamientos de cursada.
- Conceptos básicos sobre programación Inicial.
- Herramientas a utilizar.
- Instalación del software.

Clase 01

Algoritmia 1 - Introducción a la Algoritmia y Diagramas de Flujo

- Introducción al cómputo.
- Hardware y Software.
- Definición de datos, algoritmos y programas
- Características de los algoritmos.
- Diagramas de flujo.
- PSeInt

Clase 02

Algoritmia 2 - Tipos de datos y PSeInt

- Conceptos básicos de programación: variables y tipos de datos simples.
- Interfaz de PSeInt: Configuración, menús y vistas.
- Aplicación de variables en pseudocódigo.

Hardware y Software

Hardware: Se refiere a los **componentes físicos** de un sistema informático. Esto incluye dispositivos como la unidad central de procesamiento (CPU), la memoria RAM, el disco duro, la tarjeta gráfica, la placa madre, periféricos como teclados y ratones, etcétera.

Software: Son los programas y aplicaciones que al ejecutarse trabajan con los datos. Es intangible y se compone de instrucciones que le indican a la computadora cómo realizar tareas específicas. El software incluye sistemas operativos, programas de aplicación, controladores de dispositivos y cualquier otro código que permita el funcionamiento de la computadora.

¿Qué es la programación?

La **programación** es la ciencia de crear conjuntos de instrucciones para que una computadora ejecute tareas específicas, implicando la traducción de ideas y lógica humana a un lenguaje comprensible por las máquinas. Los programadores utilizan lenguajes de programación para implementar algoritmos, resolver problemas y construir software funcional, lo cual requiere habilidades lógicas, claridad estructural y eficiencia en la resolución de problemas.

Su **objetivo** es codificar instrucciones para que las computadoras ejecuten sistemas, programas y aplicaciones efectivas, accesibles y amigables para el usuario.

Conceptos vinculados con la programación

- **Lenguaje de Programación:** Es un conjunto de reglas y símbolos que permiten escribir código fuente para ser ejecutado en una computadora.
- **Código Fuente:** Es un conjunto de instrucciones escritas por un programador en un lenguaje de programación específico antes de ser traducido a un lenguaje ejecutable.
- **Sintaxis:** Son las reglas y estructuras gramaticales que rigen la forma en que se deben escribir las instrucciones en un lenguaje de programación específico, que determinan cómo se deben organizar y combinar palabras clave, operadores, variables y otros elementos dentro del código fuente. Cada lenguaje de programación tiene su propia sintaxis.

Algoritmia

La **algoritmia** es la disciplina que se ocupa de la concepción, representación, diseño, análisis, implementación y aplicación de **algoritmos**.

Un algoritmo es un conjunto **finito y ordenado** de pasos o reglas **bien definidas** que describen la secuencia de operaciones necesarias para realizar una tarea o resolver un problema específico.

La algoritmia es fundamental en el campo de la informática y juega un papel crucial en el desarrollo de software y en la resolución eficiente de problemas computacionales.

Algoritmos fuera de la computadora

Además de los **algoritmos computacionales**, que involucran computadoras y definen los procesos para dar soluciones a problemas mediante operaciones lógicas, existen algoritmos para resolver problemas de la vida cotidiana.

Al ser una **secuencia lógica de pasos ordenados** para resolver un problema, podemos encontrar algoritmos que indiquen cómo cocinar una comida, organizar un viaje, enviar un mail, lavarse los dientes, etc.



Componentes de un algoritmo

- **Entrada:** información que recibe algoritmo. Es el insumo con el que trabaja para ofrecer la solución.
- **Proceso:** conjunto de pasos que realiza el algoritmo con los datos de entrada para obtener la solución.
- **Salida:** resultados obtenidos a partir de la transformación de los valores de entrada durante el proceso.



Características de los algoritmos

Los algoritmos deben ser:

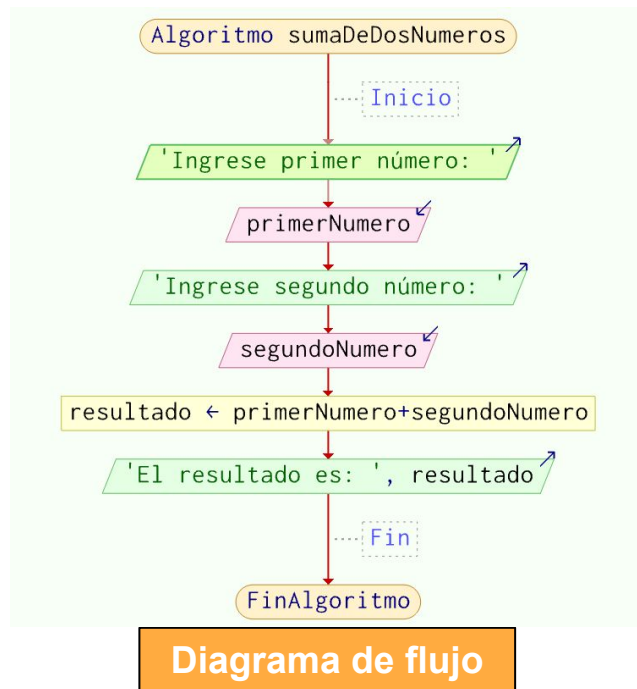
- **Claros y precisos:** no deben presentar ambigüedades.
- **Finitos:** tienen principio y un fin, un número determinado de pasos.
- **Definidos:** Los pasos o reglas deben estar claramente definidos para cada operación.
- **Entrada y salida:** Los datos de entrada deben estar claramente definidos y debe producir una salida esperada específica.
- **Ordenados:** presentan una secuencia de pasos para llegar a la solución.

Formas de los algoritmos

Existen tres maneras formales de representar un algoritmo:

- **Diagrama de flujo:** representan los algoritmos de manera *gráfica*, normalmente mediante el lenguaje UML (Lenguaje Unificado de Modelado).
- **Pseudocódigo:** describe a los algoritmos mediante un *lenguaje de alto nivel*, que luego puede traducirse a código en cualquier lenguaje de programación.
- **Código fuente:** son *una serie de instrucciones secuenciales*, escritas en un lenguaje de programación determinado, que puede ser ejecutado en una máquina a través de un compilador o intérprete.

¿Cómo se representan los algoritmos?



```
1 Algoritmo sumaDeDosNumeros
2 // Inicio
3 Escribir "Ingrese primer número: "
4 Leer primerNumero
5 Escribir "Ingrese segundo número: "
6 Leer segundoNumero
7 resultado = primerNumero + segundoNumero
8 Escribir "El resultado es: ", resultado
9 // Fin
10 FinAlgoritmo
```

Pseudocódigo

```
1 # Inicio
2 primerNumero = int(input("Ingrese primer número: "))
3 segundoNumero = int(input("Ingrese segundo número: "))
4 resultado = primerNumero + segundoNumero
5 print("El resultado es:", resultado)
6 # Fin
```


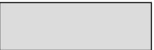
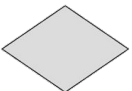



Código fuente

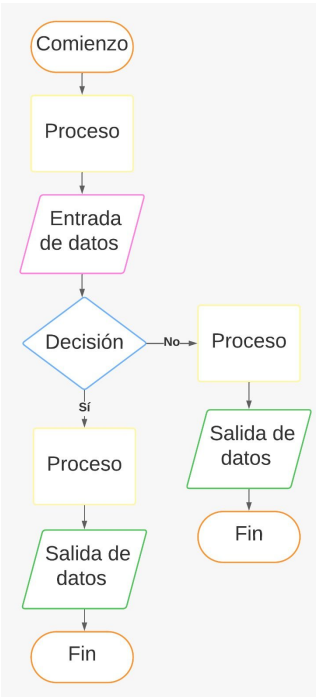
Diagrama de flujo

Es una herramienta gráfica para representar los **algoritmos**. Se utiliza a la hora de diseñar un programa. Este método visual facilita la comprensión de los pasos y la lógica del proceso. Se pueden crear fácilmente sobre un simple papel, o utilizando herramientas especializadas de software.

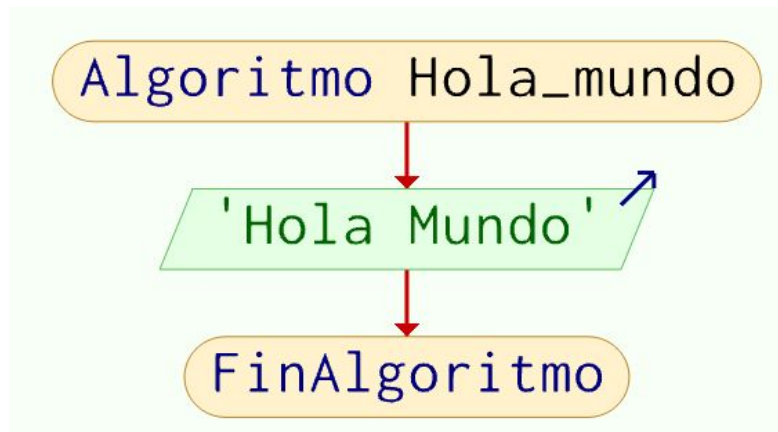
Normalmente se usa el **lenguaje UML** (*Lenguaje Unificado de Modelado*), un estándar de la industria para visualizar, especificar y construir la estructura y el comportamiento de un sistema de software.

Símbolos UML comunes en los diagramas de flujo

Forma	Nombre y descripción
	Terminador (Inicio/Fin): Indica el inicio o fin del proceso.
	Proceso: Indica una operación o tarea a realizar.
	Decisión: Un rombo indica una pregunta o condición que lleva a diferentes caminos según la respuesta (verdadero o falso)
	Conector: Conecta partes del diagrama que están en diferentes lugares.
	Flecha de dirección: Indica la dirección del flujo, mostrando la secuencia de pasos.
	Entrada / Salida de datos: Un paralelogramo indica la entrada o salida de datos.

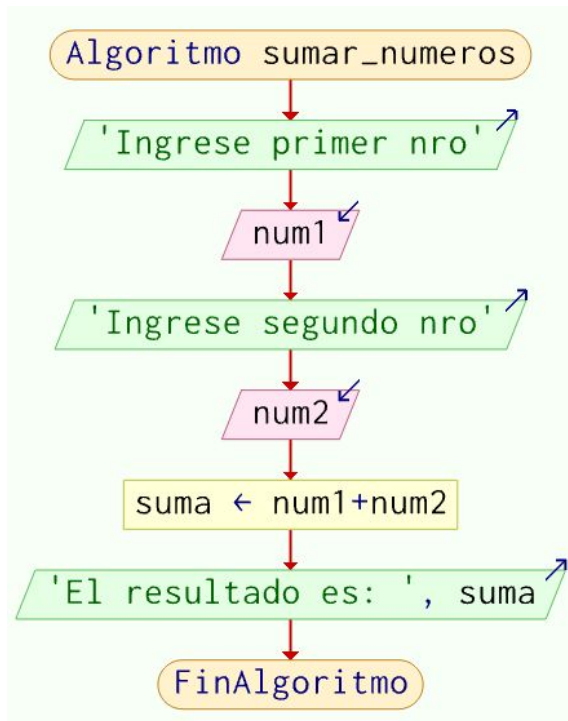


Ejemplo en PSeint



Este sencillo algoritmo, representado mediante un diagrama de flujo, imprime por consola la frase **“Hola Mundo”**.

Ejemplo en PSeint



Este algoritmo recibe dos datos:
num1 y **num2**.

Luego realiza la suma y almacena el resultado en **suma**.
Finalmente muestra en pantalla el valor de **suma**.

La lectura del diagrama de flujo que representa el algoritmo es **secuencial**, desde arriba hacia abajo.

Metodología para la resolución de problemas

La metodología para la resolución de problemas en programación implica un enfoque sistemático y estructurado. Comienza por comprender claramente el problema, descomponiéndolo en pasos más pequeños y manejables. Luego, se diseñan algoritmos para cada paso, eligiendo las estructuras de datos y algoritmos adecuados. La implementación se realiza cuidadosamente, seguida de pruebas exhaustivas para asegurar la funcionalidad y detectar posibles errores. Finalmente, se realiza la optimización, buscando mejorar la eficiencia del código. Este proceso iterativo y lógico ayuda a abordar problemas de programación de manera efectiva y eficiente.

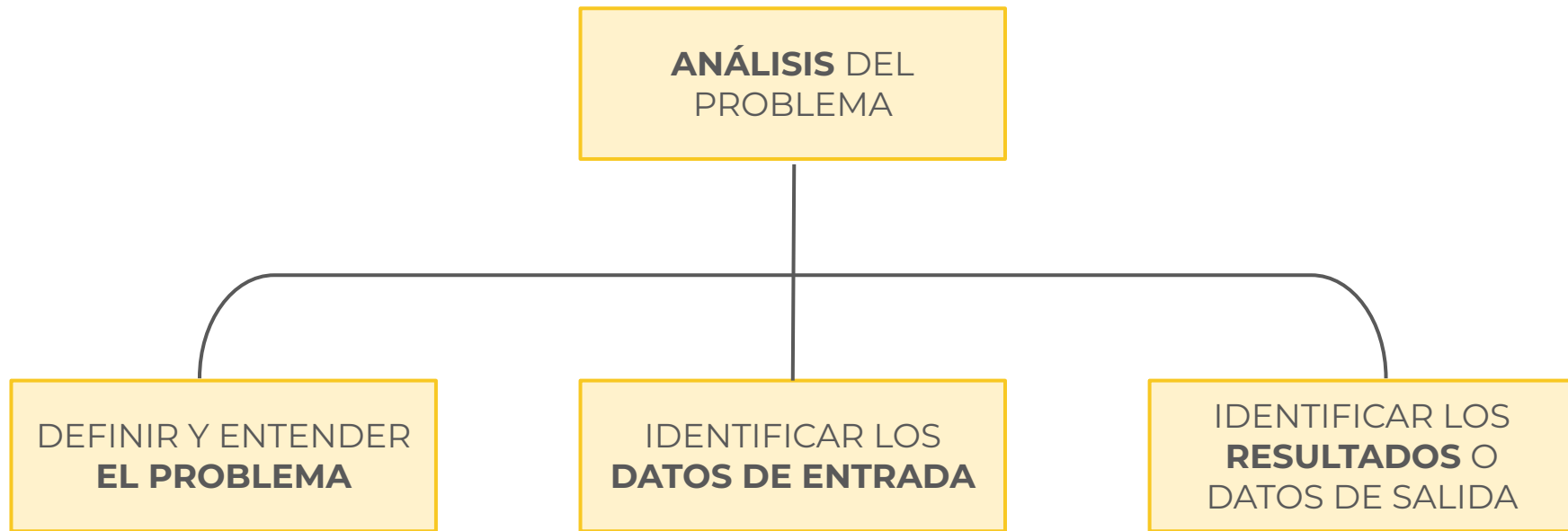
Análisis y resolución de Problemas

Utilizar una metodología apropiada para resolver problemas nos ayuda a conseguir un algoritmo eficiente y eficaz. Por lo general, los pasos a seguir son los siguientes:

1. Analizar el problema
2. Diseñar el algoritmo
3. Codificar
4. Depurar el programa

Siguiendo estos pasos correctamente se logra realizar cualquier algoritmo.

Esquema conceptual para el análisis del problema



Programa informático

Un **programa informático** es una secuencia de instrucciones escritas en un **lenguaje de programación** que manipulan un conjunto de **datos** para resolver un problema o cumplir una función determinada.

La creación de un programa informático implica la **traducción de un algoritmo o diagrama de flujo**, concebido para resolver un problema, a un conjunto de instrucciones escritas en algún lenguaje de programación, para ser ejecutadas en un dispositivo programable (generalmente, una computadora).

Estructura de un programa

La estructura de un programa se refiere a la **organización y disposición lógica de sus componentes**. En programación, la estructura de un programa típicamente incluye un encabezado, declaración de variables, entrada y salida de datos, estructuras de control, etcétera.

La estructura ayuda a organizar y comprender el código de manera más clara y mantenible. Sigue un flujo lógico que facilita la lectura y comprensión del programa. La forma específica puede variar según el lenguaje de programación utilizado, pero estos elementos son comunes en muchos programas.

Partes de un programa

En general, los programas informáticos constan de dos partes:

- **Bloque de declaraciones:** en este se detallan todos los objetos que utiliza el programa (constantes, variables, archivos, etc).
- **Bloque de instrucciones:** un conjunto de acciones u operaciones necesarias para conseguir los resultados esperados. Aquí reside la lógica del programa, donde se implementan los algoritmos diseñados para abordar el problema en cuestión.

El programa se construye mediante el uso del lenguaje de programación, transformando los algoritmos en instrucciones ejecutables.

¿Qué es un dato?

En sentido general, un dato es una pieza de información. **Es la unidad mínima de información.** En programación, un dato es esencialmente una pieza de información que almacenamos en algún sitio (por ejemplo, la memoria RAM o un pendrive). Los datos pueden ser de distintos tipos.

Independientemente de la naturaleza exacta de un dato, se almacena en la computadora como una serie de ceros y unos (utilizando el llamado “sistema binario”). Sin embargo, este proceso generalmente es “transparente” para el programador o el usuario.

Tipos de datos

Veremos que existen diferentes **tipos de datos**. El tipo indica la naturaleza del valor que se debe procesar. Se puede definir qué clase o rango de valores acepta y qué operaciones se pueden realizar con ese valor. Algunos tipos comunes son:

- **NUMEROS ENTEROS**: por ejemplo, 12, 24, 65, 1044.
- **NUMEROS REALES** (decimales y fraccionarios): 1,25; 3,1415; 1,4
- **CADENAS DE CARACTERES**: “Hola, Mundo!”; “Lunes 19 de Julio”.
- **LÓGICOS** (booleanos): Sólo pueden ser Verdadero o Falso.

Lenguaje de programación

Un **lenguaje de programación** es un conjunto de reglas y símbolos que permite a los programadores escribir instrucciones que una computadora puede entender y ejecutar. Es un medio de comunicación entre el ser humano y la máquina, facilitando la creación de software y el desarrollo de aplicaciones mediante la expresión de algoritmos y lógica de programación.

Tipos de lenguajes de programación

- **Lenguaje de máquina:** es el lenguaje que entiende la computadora de manera “nativa”. Está basado en el sistema binario (0 y 1, Falso o Verdadero). Todos los programas que escribimos son traducidos a lenguaje de máquina antes de poder ser efectivamente ejecutados.
- **Lenguaje de bajo nivel:** utiliza palabras simples que sustituyen los códigos numéricos. Estas palabras son traducidas luego a código máquina, y resulta más fácil de recordar y utilizar que el código máquina. Está más cerca del lenguaje máquina que del humano. Se lo suele llamar *Assembler* (ensamblador).

Tipos de lenguajes de programación

- **Lenguaje de alto nivel:** Es un idioma artificial, formado por signos, palabras y símbolos que permite la comunicación entre el programador y la computadora. (Ej.: Fortran, C++, Python, PHP, Java). Poseen varias ventajas:
 - Se parecen más al lenguaje humano
 - Facilitan la programación
 - Disminuyen la posibilidad de cometer errores
 - Permiten la portabilidad



PSelnt

PSelnt (abreviatura de "*Pseudo Intérprete*") es un entorno de desarrollo y aprendizaje de programación diseñado para ayudar a estudiantes a comprender los conceptos básicos de la programación y la lógica algorítmica.

PSelnt no es un lenguaje de programación en sí mismo, sino un intérprete de pseudocódigo.

Proporciona una interfaz gráfica simple y herramientas que permiten practicar la construcción de algoritmos.

La experiencia adquirida con PSelnt sienta las bases para comprender la programación en otros entornos y lenguajes.

PSeInt | Pantalla principal

The image shows the main interface of PSeInt, a programming environment for learning. The interface includes a menu bar (Archivo, Editar, Configurar, Ejecutar, Ayuda), a toolbar with various icons, and a main workspace for writing code. On the right, there is a palette of basic commands (Escribir, Leer, Asignar, etc.). At the bottom, there is a terminal window showing the output of the executed code. The code in the workspace is an algorithm to calculate the average of three numbers.

Menú.

Barra de herramientas.

Zona para escribir el código del algoritmo.

Zona con la paleta de comandos básicos.

“Terminal” o pantalla de salida de texto.

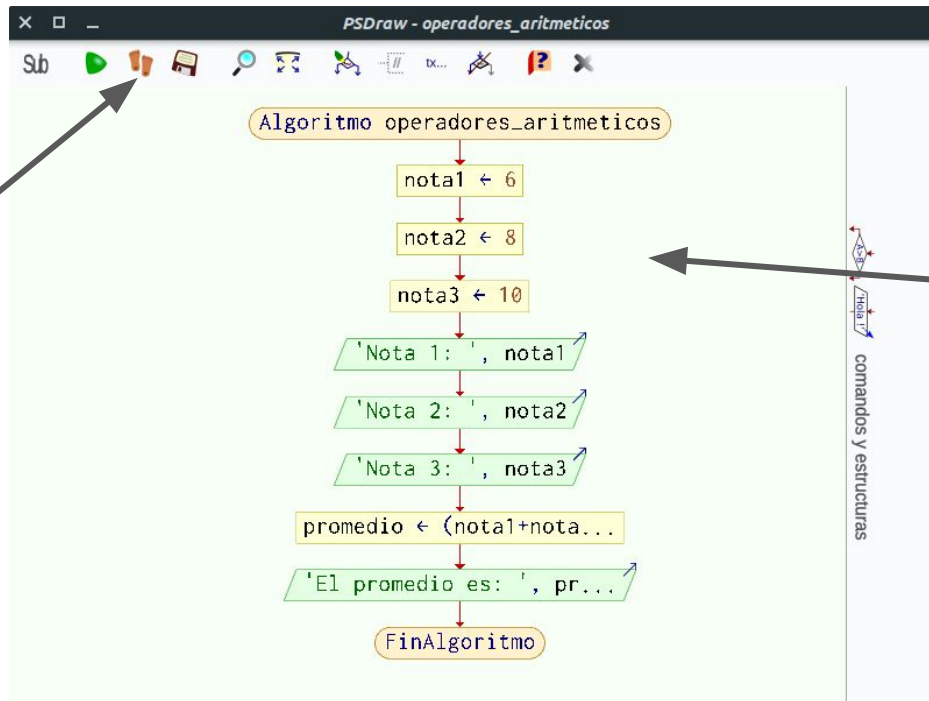
Zona de mensajes de la aplicación.

```
1 Algoritmo operadores_aritmeticos
2   nota1 = 6
3   nota2 = 8
4   nota3 = 10
5   Escribir "Nota 1: ", nota1
6   Escribir "Nota 2: ", nota2
7   Escribir "Nota 3: ", nota3
8   promedio = (nota1 + nota2 + nota3)/3
9   Escribir "El promedio es: ", promedio
10 FinAlgoritmo
11
```

*** Ejecución Iniciada. ***
Nota 1: 6
Nota 2: 8
Nota 3: 10
El promedio es: 8
*** Ejecución Finalizada. ***

La ejecución ha finalizado sin errores.

PSeInt | Pantalla PSDraw (algoritmos)



Barra de
herramientas.

Zona en la que
se ve el
algoritmo
correspondiente
al código.

Material extra

Material Extra

Artículos de interés:

- [Algoritmo: qué es, para qué sirve, ejemplos de algoritmos y cómo funciona](#) | UNIR Formación profesional
- [Algoritmos cotidianos](#) | cinconoticias.com
- [Pseudocódigo y Diagrama de Flujo: Guía Completa para Entender y Diseñar Algoritmos Efectivos](#) | Dongee

Videos:

- [¿Cómo funciona una computadora y qué hace cada parte?](#) | Nate Gentile
- [Qué es software y qué es hardware](#) | GCFAprendeLibre
- [Qué es un algoritmo y para qué se usa](#) | GCFAprendeLibre
- [¿Qué es un diagrama de flujo?](#) | Jorge Cogollo

No te olvides de dar el presente

Recordá:

- Revisar la Cartelera de Novedades.
- Hacer tus consultas en el Foro.
- Realizar el Ejercicio de Repaso.

Todo en el Aula Virtual.

Muchas gracias por tu atención.

Nos vemos pronto.