# Programación en general: el mundo de los algoritmos

## ¿Qué es un Algoritmo?

Un algoritmo es un conjunto de **instrucciones estructuradas** que tienen como **objetivo resolver un problema**. Los algoritmos están presentes en nuestra vida cotidiana en multiples campos, desde una receta de cocina que nos indica que ingredientes necesitamos, en que orden y como debemos prepararlos para obtener el plato deseado; pasando por las instrucciones de montage de un LEGO o un mueble de IKEA; una partitura musical, que indica que notas tocar para reproducir una determinada pieza o una guía de color, que nos dice, que tenemos que mezclar azul y amarillo para obtener verde.

Estas instrucciones o algoritmos pueden representarse/implementarse utilizando diferentes lenguajes y estructuras (códigos de representación).

Por ejemplo:

- Mezclando **azul y amarillo** obtendremos **verde**. Si a ese verde le añadimos blanco sera **verde claro**, pero si le añadimos negro sera **verde oscuro**.

+blanco = verde claro

- Azul + Amarillo = Verde <

+negro = verde oscuro

+ [ ] = [ ]

- [ ] + [ ] = [ ] <

+ [ ] = [ ]

En computación, los programas son algoritmos escritos en un **lenguaje** especifico y comprensible para la **máquina**, ya que, es la maquina la que tiene que realizar esas acciones.

Al igual que en el ejemplo anterior, en el que hemos escrito con tres **códigos de representación** diferentes el algoritmo que resuelve el problema '¿cómo crear color verde?', en la programación pasa igual, y existen múltiples estructuras, lenguajes y entornos, que nos permitirán implementar el algoritmo que solucione nuestro problema.

Es muy importante tener siempre en mente que EN PROGRAMACIÓN NO HAY UN SOLO CAMINO CORRECTO :)
Programar es un **proceso creativo y constructivo,** en el que, de diferentes formas, se pueden obtener resultados equivalentes. Aunque siempre podrá haber diferencias en los costes, tanto de tiempo como de memoria.

Existen muchos lenguajes de programación diferentes y aunque unos y otros frecuentemente tienen elementos en común, **cada lenguaje** se rige por sus **reglas específicas**. Esto implica que un comando o elemento que funciona en un lenguaje puede no funcionar en otro, o funcionar de otra manera; por lo que tendremos que conocer y tener en cuenta, en cada caso, los códigos de representación del lenguaje y el entorno en el que estemos programando.

Ejemplo de cómo imprimir en 4 lenguajes diferentes la misma frase "Hello World":

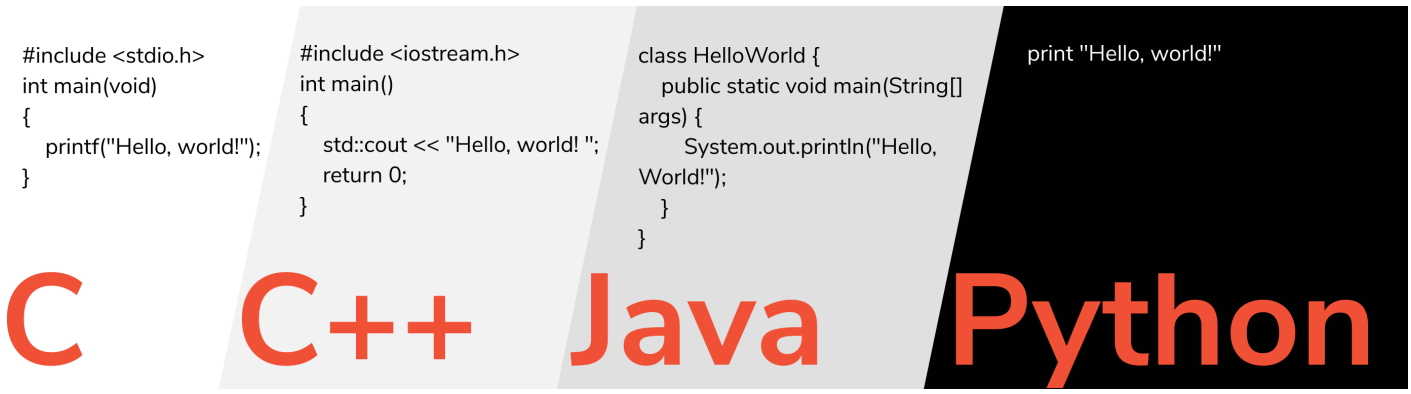


Figura 1. "Hello World" en cuatro lenguajes diferentes (C, C++, Java, Python)

> Ejercicio 1 : Busca 2 ejemplos de algoritmos en tu vida cotidiana y represéntalos con dos códigos diferentes.

# ¿Qué es la programación?

La **programación informática,** se refiere al proceso de construcción de estos grupos de instrucciones, para que sean procesadas por un ordenador o microcontrolador. La programación, se basa en flujos de datos que se almacenan, comprueban y transforman de manera estructurada.

En un sistema computacional vamos a tener elementos de dos tipos: el **software,** que es la parte inmaterial, y el **hardware,** que es la parte material.
El **software,** son los componentes lógicos ("Software", 2022), las instrucciones o algoritmos, es decir, los programas que serán seguidos por el hardware. El **hardware,** son los componentes físicos que constituyen el ordenador/computadora (disco duro, tarjeta gráfica, pantalla, ratón, altavoz, cámara ...). El software, le dice al ordenador lo que tiene que hacer, para lograr el resultado deseado. El software es el ADN de la máquina. No obstante, pese a que esta distinción está ampliamente extendida, en realidad, no está tan claro dónde debe situarse la línea que establece la diferencia entre hardware y software.

Lo que sí está claro es que esta combinación de hardware y software constituye el sistema, es decir, la infraestructura que permitirá que los datos introducidos en un programa, a los que se conocen como **inputs/inlets/entrada**, sean procesados, es decir, que sean comprobados, leídos, almacenados, modificados.

El resultado que este programa proporciona se conoce como **output/outlet/salida.** Esta terminología y estructura (entrada -> proceso -> salida), se aplica también, a los procesos internos de los programas, ya que, un programa está constituido por otros programas, que son, multiples procesos conectados e interrelacionados.
Esta idea de entrada y salida de datos, nos va a acompañar durante todo el curso, ya que, en nuestros proyectos vamos a trabajar con un conjunto de datos de entrada, que nos permitirán obtener un resultado o salida. Generalmente, todo proceso tiene una entrada de datos (input), un procesamiento, durante el cual se realizan acciones con esos datos, y una salida de datos(output) que es el resultado del proceso. Programar es la acción de construir esos programas para ordenador (Scratch, 2022).

Para explicar y hacer más comprensible cómo funcionan los algoritmos, suelen emplearse metáforas. Así, es posible establecer similitudes entre algo tan abstracto, como es la programación, con procesos que nos resultan más familiares. Vamos a tomar como **metáfora** los sistemas hidrográficos: ríos, arroyos, pozos, lagos, meandros, y las infraestructuras que el ser humano introduce en estos sistemas fluviales para modificar el curso y estado del agua: presas, centrales hidroeléctricas, sistemas de canalización para el riego, acueductos o simples marcas en la pared de un canal, para medir el nivel del agua y obtener información acerca del crecimiento de los ríos.


Figura 2. Mapa de embalses de la cuenca del Ebro.

Por ejemplo:

En Zaragoza el nivel del agua del rio se mide una vez (Proceso 0: contar) todos los días (Proceso 1: comparar) y se apunta en una libreta (Proceso 2: almacenar). Este nivel, ha de ser siempre superior a 50 (Proceso 3: comparar). Cuando el nivel de agua en Zaragoza, desciende por debajo de 50, se avisará al embalse (Proceso 4.1.1: enviar información) para que deje salir más agua (Proceso 4.1.2: cambiar), si el nivel de agua no ha bajado por debajo de 50, no se avisara al embalse y el agua que deja salir este, seguirá siendo la misma (Proceso 4.2: no cambiar). Nombre descriptivo de este programa: Comprobación del estado del agua en Zaragoza, y estado del embalse en función de esa comprobación.

El agua de este sistema hidrográfico va a ser los datos. Y las infraestructuras introducidas por el ser humano para controlar estas aguas van a ser los programas que procesaran estos datos. De esta manera, iremos encadenando procesos por los que los datos pasan para conseguir nuestro objetivo.

> Ejercicio 2: Identifica y etiqueta los diferentes procesos que conforman los algoritmos que has elegido para el ejercicio 1. Por etiqueta, nos referimos a que clasifiques estos procesos como en el ejemplo anterior, en función del tipo de acción que se realiza en cada proceso, utilizando categorías como comparar, almacenar o cualquier otra que se te ocurra.

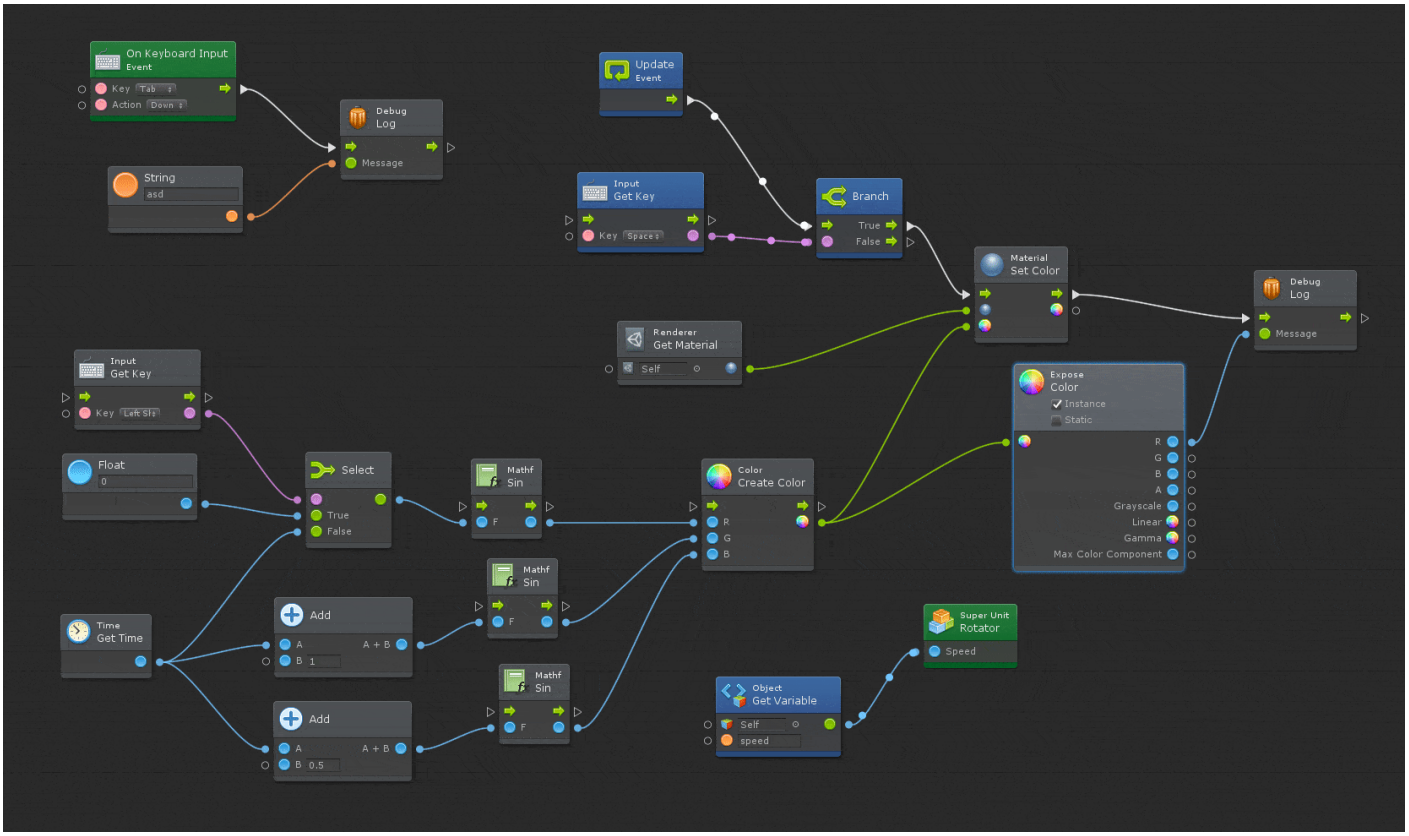


Figura 3. Un gráfico de un lenguaje de programación visual llamado "Bolt" en funcionamiento.

En la Figura 3. podemos ver un ejemplo animado de **programación gráfica**, que visualmente, se asemeja al mapa de la cuenca hidrográfica del Ebro de la Figura 2. Vemos cajitas conectadas por unas líneas, a través de las cuales, unas bolitas se desplazan de una caja a otra. Cada una de las cajitas representa un proceso, y las bolitas son los datos que entran por la parte izquierda de una cajita, en la cajita son procesados, y salen por la parte derecha en dirección a otra cajita. La dirección del desplazamiento de las bolitas, indica la dirección de los datos a través de estos procesos, que en este caso fluyen de izquierda a derecha, y es en este orden en el que se procesan.

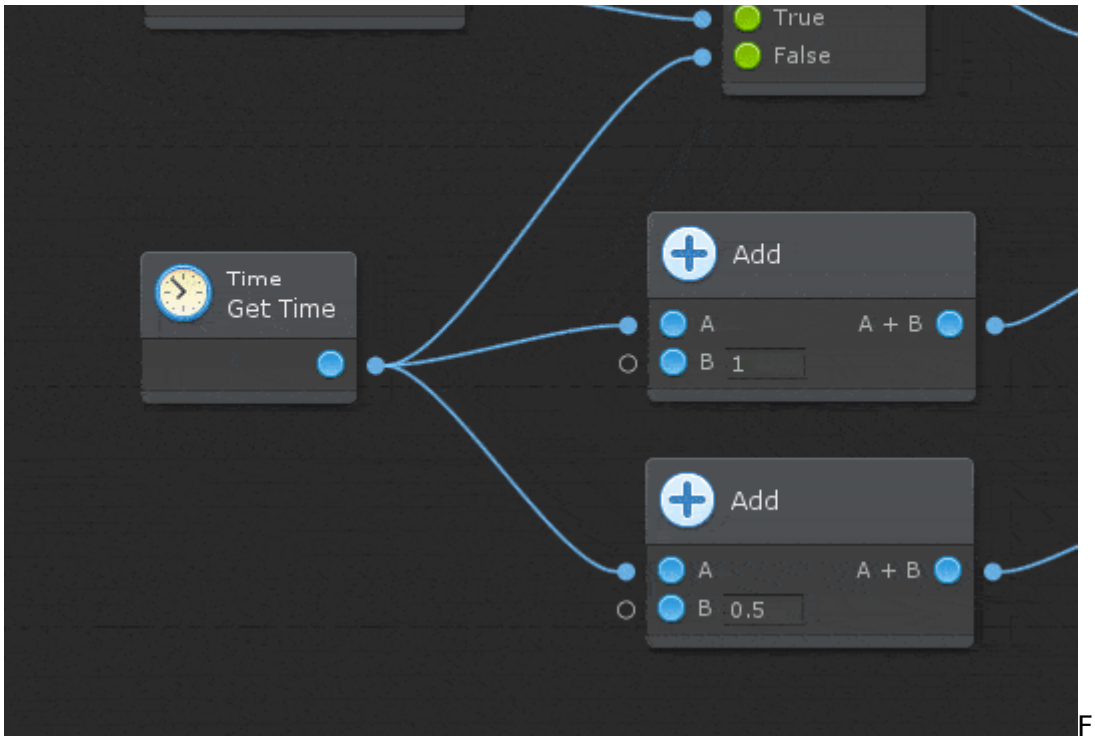

Figura 4. Detalle de la figura 3. que muestra un mismo outlet puede alimentar los inlets de varios procesos (el mismo outlet de Time hace de input para Select, Add y Add) y un proceso puede tener varios inlets (Select tiene 3 inlets).

Vemos que la salida de una sola cajita, puede estar conectada con las entradas de varias cajitas, ya que, el mismo resultado de un proceso puede alimentar varios procesos (Cajita Time). Y que, un proceso puede tener varios datos de entrada diferentes, que proceden de diferentes procesos previos (Cajita Select).