**Turing, Alan**: Matemático inglés (1912-1954) que estableció las bases de la computación moderna. En el campo de la inteligencia artificial creo la prueba de Turing (1950), un concepto mediante el cual se puede establecer si una maquina exhibe un comportamiento inteligente igual al de un ser humano (no si es inteligente o no, esto sería más difícil de responder).

**Prueba de Turing**: Con esta prueba se evalúa el si una máquina es capaz de exhibir un comportamiento inteligente similar o indistinguible del de un ser humano. En esta prueba, un evaluador humano a través de preguntas realizadas en lenguaje natural a dos evaluados, uno es una máquina y otro es un humano, tiene que ser capaz de distinguir si quien le responde es la máquina o el ser humano. Para ello se basa sólo en las respuestas escritas.

**Inteligencia Artificial**: Se puede definir como un conjunto de algoritmos que tiene como fin crear maquinas que presentan las mismas capacidades que el ser humano. Coloquialmente, el término "inteligencia artificial" se usa a menudo para describir máquinas (o computadoras) que imitan funciones "cognitivas" que los humanos asocian con la mente humana, como "aprender" y "resolver problemas". A medida que las máquinas se vuelven cada vez más capaces, las tareas que se considera que requieren "inteligencia" a menudo se eliminan de la definición de IA, un fenómeno conocido como efecto de IA. Una broma en el teorema de Tesler dice que "la IA es lo que no se ha hecho todavía".

Se pueden distinguir varios tipos de Inteligencia Artificial:

Sistemas que piensan como humanos: Toman decisiones, razonan, aprenden.Redes Neuronales.

Sistemas que actúan como humanos: Robots.

Sistemas que piensan racionalmente: Sistemas que imitan el pensamiento lógico racional de los humanos. Sistemas expertos.

Sistemas que actúan racionalmente: idealmente, son aquellos que tratan de **imitar de manera racional el comportamiento humano,** como los agentes inteligentes

**Programación**: en el ámbito de la informática mediante la programación transformamos en código o lenguaje de programación una serie de instrucciones para qué un determinado sistema realice las tareas que el usuario solicita.

**Lenguaje de programación**: **es un idioma artificial prediseñado formado por signos, palabras y símbolos** que permite la [comunicación](https://concepto.de/comunicacion/) entre el programador y el ordenador.

**Lenguaje máquina:** El primer lenguaje que se utilizó para comunicarse con una máquina, es muy básico y limitado basado en el [sistema binario](https://concepto.de/sistema-binario/) (uso de los números 0 y 1 en distintas combinaciones) que es el lenguaje que los ordenadores reconocen, por lo que aún hoy todo lenguaje es convertido a este. Fue reemplazado, ya que resultaba una forma de programación tediosa y difícil.

**Lenguaje ensamblador:** Es un tipo de lenguaje que surgió después del lenguaje máquina, estos lenguajes hacían uso de códigos de palabras. Se utilizaban palabras simples, mnemotécnicas y abreviaturas que tenían su correlativo y eran traducidas al código máquina. El lenguaje ensamblador resultaba más fácil de recordar y realizar por el usuario que el código máquina.

**Lenguaje de alto nivel:** A finales de la década del 50 surgió el Fortran, un lenguaje de programación desarrollado por IBM que dio inicio a la aparición de lenguajes basados en conjuntos de algoritmos mucho más complejos. Estos lenguajes se adaptaban a distintos ordenadores y eran traducidos por medio de un software al lenguaje de máquina.

**Programación**: en el ámbito de la informática refiere a **la acción de crear programas o aplicaciones** a través del desarrollo de un [código fuente](https://concepto.de/codigo-fuente/), que se basa en el [conjunto](https://concepto.de/que-es-un-conjunto/) de instrucciones que sigue el ordenador para ejecutar un [programa](https://concepto.de/programa-informatico/).

**Programación estructurada**: Busca mejorar y reducir el tiempo del proceso al utilizar subrutinas (subalgoritmos dentro del algoritmo principal que resuelve una tarea).

**Programación modular**. Divide los programas en módulos para trabajar con ellos y resolver los problemas de manera más simple.

**Programación orientada a objetos**. Utiliza objetos (entes con características, estado y comportamiento) como elementos fundamentales para la búsqueda de soluciones.

**Transistor**: Componente electrónico semiconductor de tamaño nanométrico que permite convertir una señal analógica en una señal digital. Los transistores normalmente se incluyen dentro de un circuito integrado, los cuales, a su vez, constituyen los circuitos electrónicos que forman parte de los ordenadores. Los transitores se pueden utilizar para dos tareas, para amplificar la señal eléctrica (función de amplificación) o para dejar pasar o bloquear la corriente (función de interruptor). https://www.areatecnologia.com/TUTORIALES/EL%20TRANSISTOR.htm

**Puertas lógicas**: Una puerta lógica es un **bloque en la construcción de un circuito digital**. La mayoría de las puertas lógicas **tienen dos entradas y una salida y están basadas en álgebra booleana**. En un momento dado, cada terminal está en una de las dos condiciones binarias false (alto) o true (bajo). Falso representa 0, y verdadero representa 1. Dependiendo del tipo de puerta lógica que se utilice y de la combinación de entradas, la salida binaria será diferente. Una puerta lógica puede ser pensada como un interruptor de luz, donde en una posición la salida está apagada-0, y en otra, está encendida-1. Las puertas lógicas son comúnmente usadas en circuitos integrados (IC). https://descubrearduino.com/puertas-logicas/

**Puerta AND:** Con este operador la única manera de que la señal de salida sea 1, es que las dos señales de entrada sean 1. Para el resto de combinaciones, la señal de salida es 0. Actúa de la misma forma que el operador lógico “y”.

**Puerta OR**: Con este operador la única manera de que la señal de salida sea 0, es que las dos señales de entrada sean 0. Actúa de la misma forma que el operador lógico “o”

**Puerta XOR:** También llamada “eXclusive-OR”. Esta puerta funciona como el operador lógico “uno u otro”. Para que la señal de salida sea 1, las dos señales de entrada tienen que ser diferentes.

**Puerta NOT:** Este operador es un inversor lógico, tiene sólo una entrada. Si la señal de entrada es 1, la de salida es 0. Si la señal de entrada es 0, la señal de salida es 1

**Puerta NAND:** Este operador actúa como una puerta NAND seguida de una puerta NOT. Funciona como la operación lógica “y” seguida de negación. La señal será 0 cuando las dos señales de entrada sean 1. En el resto de los casos, la señal de salida es 1.

**Puerta NOR:** Este operador actúa como la puerta OR seguida de la puerta NOR. El único caso en que la salida es 1, es si las dos entradas son 0.

**Puerta NXOR:** Es la negación de la puerta XOR. Para que la señal de salida sea 1, las dos señales de entrada tienen que ser iguales.

**Lógica proposicional**, o a veces lógica de orden cero, es un sistema formal cuyos elementos más simples representan proposiciones, y cuyas constantes lógicas, llamadas conectivas lógicas, representan operaciones sobre proposiciones, capaces de formar otras proposiciones de mayor complejidad.

**Algebra:** es la rama de la matemática que estudia la combinación de elementos de estructuras abstractas acorde a ciertas reglas. Originalmente esos elementos podían ser interpretados como números o cantidades, por lo que el álgebra en cierto modo originalmente fue una generalización y extensión de la aritmética.

**Algebra booleana**: Se denomina así en honor a George Boole (1815-1864), matemático inglés autodidacta, que fue el primero en definirla como parte de un sistema lógico. El álgebra de Boole fue un intento de utilizar las técnicas algebraicas para tratar expresiones de la lógica proposicional.

**Ley de Moore**: Escrita por el ingeniero Gordon Moore en 1965. Este ingeniero y co-fundador de Intel, fue el primero que se dio cuenta que el número de transistores por unidad de superficie se duplicaba cada año. En 1975 corrigió su propia ley, que ahora dice que el número de transistores por unidad de superficie, MTr/mm2 (millones de transistores por milímetro cuadrado) se duplica cada dos años. Es decir, cada dos años el tamaño de los transistores disminuye a la mitad. En la actualidad (2020) los transistores de Intel han alcanzado los 10 nm, con lo que en 1 mm2 caben alrededor de 100 millones de transistores.

**Litografia 10 nm:** Termino relacionado con el tamaño de los transistores que integran un chip. Una litografia de 10 nm, quiere decir que se usan transistores de 10 nm, cuanto más pequeño sea el tamaño del transistor, mas unidades entran en un microchip, mejorando eficiencia, rendimiento y generando menos calor.

**Bit**: Unidad más pequeña de información a la que se traduce cualquier lenguaje de programación para que el ordenador lo pueda entender. Esta información puede ser un 1 o un 0, es decir, encendido o apagado.

**Byte**: Conjunto de 8 bits, que se utiliza para expresar más información, como en el caso de los bits, sólo utiliza 1 y 0, pero dependiendo del lugar que ocupan toman diferente valor. Si pusiéramos estos dígitos en 8 casillas, cada una de ellas es una potencia de 2, la de la derecha sería 2E0, la siguiente 2E1, la siguiente 2E2…, es decir la más a la derecha tendría un valor de 1, la siguiente de 2, la siguiente de 4 y así sucesivamente. Se pueden hacer más operaciones sumando las casillas que estén “encendidas”, si todas estuvieran “encendidas”, la suma daría un total de 255 (2E8-1)

**Circuito electrónico:** Es un conjunto de componentes que permiten transmitir la electricidad y convertirla en luz, sonido, movimiento..etc. Son la base de la tecnología moderna.

**Circuito digital**: En este caso las ondas eléctricas son convertidas en ceros y unos y esta información se guarda en la memoria **RAM “**Random Access Memory**”**, mientras el circuito tenga energia, o en el disco duro, haya o no energía. La unidad mínima de información es un bit, y 8 bits hacen un byte. El procesador puede hacer operaciones con los bytes y convertirlos en otros mediante instrucciones, que no son más que código de programación.

**Periférico**: En informática un periférico es cualquier aparato o dispositivo independiente conectado al procesador central de un ordenador. Nos permite comunicarnos con el ordenador.

**Periférico de entrada**: Dispositivos que proporcionan información de cualquier tipo(pueden ser dígitos, palabras, imágenes..) a la unidad de procesamiento central del ordenador. Los más conocidos serían, el teclado, el ratón, el micrófono, una webcam. El procesador conmuta los datos proporcionados y devuelve al usuario a través de los dispositivos periféricos de salida.

**Periférico de salida**: Dispositivos que devuelven los resultados del procesamiento de los datos al usuario, estos datos pueden ser números, sonidos, imágenes. Algunos ejemplos de dispositivos de salida son los monitores, altavoces, impresoras.

**Periférico de entrada/salida**: Dispositivos tanto para que el usuario envíe información a la CPU, como para recibir la información de ella. Pantallas táctiles o los USB son dos ejemplos.

**CPU**: Central Processing Unit, el cerebro del ordenador. Tradicionalmente, se compone de la **ALU**, “Arithmetic Logical Unit”, los **registros de memoria**, que suministran los datos a la ALU y almacenan los resultados y la **CU** o “Control Unit”, que se encarga de acceder a las instrucciones situadas en la memoria principal, interpretarlas y comunicarse con los registros y la ALU y otros elementos, para que cumplan su función de manera coordinada. **https://en.wikipedia.org/wiki/Central\_processing\_unit#Microprocessors**

**ALU**, “Arithmetic Logical Unit”, circuito digital dentro del procesador que se encarga de realizar operaciones aritméticas y lógicas. Se comunica con los registros que le envían los datos y la CU que le indica qué operación realizar.

**Registros de memoria**, un tipo de memoria muy rápida y de poca capacidad, que suministra los datos a la ALU y almacenan los resultados.

**CU** o “Control Unit”, unidad en el interior de la CPU que dirige el funcionamiento del procesador, controla la ALU, los registros, la memoria y la interfaz I/O. Dirige el flujo de datos entre la CPU y otros componentes.

**Microprocesadores**: Hoy en día las CPUs de la mayoría de ordenadores son microprocesadores, es decir todos los elementos se encuentran integrados en un chip.

**Bus**: Sistema de comunicación entre los distintos componentes del ordenador

**Arquitectura-von-neumann**: El rasgo más importante de este tipo de arquitectura de ordenadores, es que los datos y las instrucciones comparten el mismo bus. Por lo que no se puede acceder a una instrucción y manejar un dato a la vez. Esto afecta a la velocidad de procesamiento y se conoce como “von Neumann bottleneck”. En los ordenadores modernos se ha conseguido mitigar esta limitación mediante el uso de caches (pequeñas memorias de acceso muy rápido) entre la CPU y la memoria central, en concreto, los caches más cercanos a la CPU, están divididos en caches de datos y de instrucciones (split cache) y utilizan diferentes buses. La ventaja es que las instrucciones son tratadas como datos, esto hace que los programas puedan escribir programas. Hace posible los ensambladores, los compiladores y otras herramientas.

**Arquitectura Harvard**: A diferencia de la Arquitectura von-Neumann, en esta disposición los datos y las instrucciones se almacenan en sets de direcciones de memoria diferentes (todos dentro de la memoria RAM pero separados) y no comparten buses. Se pueden escribir o leer de la memoria y a la vez acceder a una instrucción.

**Hardware**: Partes físicas, tangibles de un ordenador: microprocesador o CPU, placa madre, memoria RAM, disco duro, fuente de alimentación, tarjeta gráfica, carcasa, dispositivos periféricos.

1. [Microprocesador](https://techlandia.com/definicion-microprocesador-sobre_99685/) o unidad central de procesamiento (CPU)
2. [Placa o tarjeta madre](https://techlandia.com/funcion-placa-madre-computadora-sobre_168309/)
3. La memoria de acceso aleatorio o [memoria RAM](https://techlandia.com/utiliza-memoria-ram-hechos_444633/)
4. [Dispositivo de almacenamiento](https://techlandia.com/cuales-son-dispositivos-almacenamiento-datos-sobre_349592/) de datos o disco duro
5. Fuente de poder o [fuente de alimentación](https://techlandia.com/cuales-son-distintas-partes-fuente-alimentacion-computadora-hechos_93809/)
6. [Tarjeta gráfica](https://techlandia.com/definicion-tarjeta-grafica-hechos_172986/) u otras placas opcionales (GPU)
7. Carcasa, caja, “case” o “chasis”
8. [Dispositivos periféricos](https://techlandia.com/tipos-dispositivos-entrada-sobre_37283/): Mouse, teclado

**Software**: Partes intangibles de un ordenador. Por ejemplo, el sistema operativo y los distintos programas o ampliaciones.

**Sistema operativo (OS):** Es el software que, se comunica con el *hardware* y gestiona sus recursos, además provee de herramientas al resto de aplicaciones del sistema para que puedan funcionar. Tiene privilegios de ejecución sobre el resto de programas.

**BIOS:** El BIOS fue creado en 1975, y sus siglas significan Basic Input Output System o sistema básico de entrada y salida. Su función principal es la de **iniciar los componentes de *hardware* y lanzar el sistema operativo** de un ordenador cuando lo encendemos

**Firmware**: Es un programa muy básico que controla los circuitos electrónicos de un dispositivo. Se almacena en una memoria ROM especial, en un ordenador está integrado en la placa base. El firmeware es un código con instrucciones para decirle al hardware como tiene que funcionar. También se conoce como soporte lógico inalterable.

**Raspberry pi**: Se trata de un ordenador de bajo coste. El Rasperry pi4 cuenta con un procesador, una memoria RAM de 2, 4, u 8 Gigas, puertos USB, HDMI, conexión Ethernet, wireless o bluetooth . Se le pueden conectar distintos periféricos, teclado, ratón, pantalla. Se le pueden instalar diferentes sistemas operativos, según el uso que se le quiera dar.

**Arduino:** El proyecto arduino nació en 2003 en Italia, de mano de unos estudiantes con idea de crear una placa de electrónica de bajo coste, para promover el acceso y uso de la electrónica y la programación y con la filosofía de “learning by doing”. Arduino es una plataforma de creación electrónica y de código abierto, basada en hardware y software libre, para que cualquier persona o empresa pueda crear su propia placa. https://www.xataka.com/basics/que-arduino-como-funciona-que-puedes-hacer-uno

**Lenguaje de programación**: Lenguaje especial que un programador utiliza para comunicarse con una máquina. Son una serie de instrucciones que tienen que ser escritas con unos signos, palabras, sintaxis o estructura determinadas y que forman un programa.

**Lenguaje maquina**: Lenguaje de bajo nivel, el único lenguaje que puede entender un dispositivo electrónico, se trata simplemente de una colección de 0 y 1, también se llama lenguaje binario, porque sólo usa dos dígitos.

**Lenguaje ensamblador**: Primer intento de sustitución del lenguaje máquina. Es también un lenguaje de bajo nivel, pero algo más sofisticado que el lenguaje maquina sirve para pasar de un lenguaje de alto nivel a

**Lenguaje de alto nivel**: Lenguaje de programación cercano al lenguaje natural, al que utilizamos los humanos, también se llama código fuente (del que se parte). Para que una máquina lo entienda es necesario convertirlo a lenguaje máquina, para ello se usan traductores, compiladores y ensambladores. El usar uno u otro depende de en que lenguaje esté escrito el código fuente.

**Traductores**: Son programas que convierten las instrucciones escritas en otro lenguaje a lenguaje binario y la ejecutan a medida que se traducen.

**Compilador**: Es un tipo de traductor a través del cual todos las instrucciones de un programa escrito en otro lenguaje se traducen a lenguaje maquina a la vez y después se ejecutan. La entrada es una sentencia en lenguaje de alto nivel y la salida un ejecutable. Se puede almacenar el programa traducido y si se necesita no hace falta traducirlo otra vez, el lenguaje ya está compilado.

**Interprete**: Es como un compilador, solo que la salida es una ejecución. El programa de entrada se reconoce y ejecuta a la vez. No se produce un resultado físico (código máquina) sino lógico (una ejecución).

**Lenguaje-compilado:** Un lenguaje compilado se puede ejecutar por sí solo, no necesita de otra herramienta. Eso sí, solo se puede utilizar en la máquina para la que se ha compilado. Cada lenguaje compilado tiene su versión para las diferentes plataformas.

**Lenguaje-interpretado**: A diferencia de un lenguaje compilado, un lenguaje interpretado necesita de un intérprete para poder ser convertido a lenguaje máquina. Es más flexible porque se puede utilizar la misma versión en distintas plataformas, pero se necesita un interprete diferente para cada una de ellas.<https://ingenieriadesoftware.es/diferencia-compilador-interprete-transpilador/> <https://ocw.uma.es/course/view.php?id=61>

**Frontend**: En el entorno del desarrollo web, frontend se refiere a la parte de una aplicación o dispositivo a la que un usuario puede acceder. Los desarrolladore frontend trabajan para que la interfaz del usuario sea fácil de usar y atractiva. Los lenguajes más usados en frontend: HTML, CSS y JavaScript.

**HTML**: “HyperText Markup Language” (lenguaje de marcas de hipertexto) lenguaje de marcado para la elaboración de páginas web. En realidad no es un lenguaje de programación. HTML sólo sirve para indicar como va ordenado el contenido de una página web. Esto lo hace por medio de las marcas de hipertexto las cuales son etiquetas conocidas en inglés como tags. https://codigofacilito.com/articulos/que-es-html

**CSS**: Este lenguaje sirve para definir el diseño de una página web. La filosofía de CSS se basa en intentar separar lo que es la estructura del documento HTML de su presentación. Con CSS puedes crear animaciones, efectos, manejar colores, fuentes, tamaños y mucho más. En realidad no es un lenguaje de programación.

**JavaScript**: Es un lenguaje de programación que sirve para crear páginas webs dinámicas, es decir introducir animaciones, textos que aparecen y desaparecen... Es un lenguaje interpretado. Los programas escritos en JavaScript se pueden probar directamente en cualquier navegador, sin necesidad de intermediarios. No tiene nada que ver con Java.

**Backend:** Se refiere a la parte de una aplicación o dispositivo a la que el usuario no puede acceder, lo que no se ve. Además de datos que son inaccesibles al usuario, esta capa contiene la lógica que maneja dichos datos. También se encarga de enlazar con el servidor. Lenguajes relacionados: PHP, python, java, C++, lisp, R, Prolog.

**Python** Es un lenguaje muy flexible y sencillo. Es de los más usados en IA, pero también sirve para aplicaciones móviles, web, data science. Es uno de los lenguajes más utilizados para IA. Su simplicidad y las filosofías DRY (Don’t Repeat Yourself) y RAD (Rapid Application Development) en las que se basa lo convierten en un candidato ideal. Puede utilizarse tanto para estructurar datos como para generar algoritmos de IA. Además, dispone de un catálogo de librerías muy extenso que permite hacer realidad cualquier tipo de proyecto. Ningún soporte se resiste a Python, puesto que sirve para trabajar en aplicaciones de todo tipo, sean mobile, web, data science o IA.

**R**: Este lenguaje dispone de paquetes de programación muy numerosos. Algunos de ellos se utilizan en el ámbito del Machine Learning, como RODBC. Para garantizar la funcionalidad de la IA, implementan algoritmos de aprendizaje automático. Se trata de uno de los mejores lenguajes para analizar y tratar con datos. Por ello, es posible crear buenas IA con finalidades estadísticas.

**Lisp:** Desde su nacimineto en 1958 de la mano de John McCarthy, Lisp no ha dejado de crecer. De hecho, su creador trabajó en el MTI junto a Marvin Minsky, uno de los padres de la Inteligencia Artificial. Lisp trabaja con expresiones simbólicas y prototipado, herramientas útiles en el campo del Machine Learning. Además, se utiliza en proyectos como CYC, cuyo objetivo es permitir a las aplicaciones basadas en IA ejecutar razonamientos similares a los humanos.

**Prolog:** Suele utilizarse en proyectos médicos y para diseñar sistemas de IA especializados. Se trata de un lenguaje de referencia en el entorno de la ingeniería. Una de sus funcionalidades más destacadas es que permite automatizar el backtracking, que consiste en buscar errores y retroceder hasta el punto anterior para tomar otra alternativa. Para conseguir esto, se basa en estructuras de datos arbóreas que facilitan la búsqueda de patrones. Todas estas características combinadas convierten este lenguaje en uno de los más flexibles.

**Java:** es un lenguaje de programación orientado a objetos que posee todas las herramientas necesarias para trabajar en proyectos de Inteligencia Artificial. Las características más destacadas de Java son la transparencia, la mantenibilidad y la portabilidad. Permite codificar algoritmos muy fácilmente y es un lenguaje escalable. Teniendo en cuenta que una IA está basada en gran medida en estos algoritmos, Java es una muy buena opción. Además, dispone de interfaces de datos muy atractivas para mejorar la experiencia del usuario. Si tenemos alguna duda, dispone de una nutrida comunidad de usuarios que pueden ayudarnos.

**C++**: se dice que es el lenguaje de programación más rápido del mundo. En este sentido, se trata de uno de los mejores lenguajes de programación para IA, puesto que esta tecnología requiere de ejecuciones y tiempos de respuesta rápidos. C++ convierte el código creado por el usuario en datos legibles para las máquinas, cosa que lo convierte en una opción muy eficiente.

**PHP**: Un lenguaje backend del que se dice que es muy fácil de aprender, tiene programación HTML integrada, se utiliza para la creación de páginas web dinámicas. Es de código abierto y puede ser usado con cualquier sistema operativo y con cualquier servidor.

**SQL:** Por sus siglas en ingles “**Structured Query Language”**. Es un lenguaje que surgió para ser usado con las bases de datos relacionales. Sirve para interaccionar con los gestores de bases de datos. Para consultar datos, extraer datos, modificar datos, escribir o eliminar datos.

**Repositorio de control de versiones**: Es una herramienta fundamental para los programadores. Sirven para guardar un registro de todos los cambios realizados en un proyecto.

**Git:** Es un software libre creado por Linus Torvalds, es muy útil para seguir los cambios de un proyecto. Cuando queremos guardar una versión del proyecto, git puede hacer una instantánea del proyecto en ese momento y le asigna un código. Podemos seguir haciendo cambios, y cuando queremos volver a guardar el proyecto con esos cambios, decirle a git que haga otra instantánea, si en un momento dado por cualquier causa, queremos volver a una versión concreta anterior, git puede restaurarla. <https://www.youtube.com/watch?v=ANF1X42_ae4> https://git-scm.com/docs

**Branch**: Cuando hay varias personas trabajando en el mismo proyecto y que se van a encargar de partes diferentes del mismo, se pueden sacar “ramas” o flujos de trabajo del proyecto principal y cada programador trabaja en su rama, una vez finalizan, git se encarga de fusionar las ramas. Git también es capaz de detectar conflictos, es decir, si el trabajo de dos personas se solapa, nos advierte sobre el conflicto a la hora de fusionar las ramas.

**Github**: Github es por un lado un gestor de proyectos y un repositorio online de control de versiones y por otro lado una plataforma social de trabajos en grupo. Para depositar o descargar proyectos a este o desde este repositorio, se utiliza git. Github es una apliaciónde Microsoft.

**Repositorio de control de versiones**: Es una herramienta fundamental para los programadores. Sirven para guardar un registro de todos los cambios realizados en un proyecto.

**Git:** Es un software libre creado por Linus Torvalds, es muy útil para seguir los cambios de un proyecto. Cuando queremos guardar una versión del proyecto, git puede hacer una instantánea del proyecto en ese momento y le asigna un código. Podemos seguir haciendo cambios, y cuando queremos volver a guardar el proyecto con esos cambios, decirle a git que haga otra instantánea, si en un momento dado por cualquier causa, queremos volver a una versión concreta anterior, git puede restaurarla. <https://www.youtube.com/watch?v=ANF1X42_ae4> https://git-scm.com/docs

**Branch**: Cuando hay varias personas trabajando en el mismo proyecto y que se van a encargar de partes diferentes del mismo, se pueden sacar “ramas” o flujos de trabajo del proyecto principal y cada programador trabaja en su rama, una vez finalizan, git se encarga de fusionar las ramas. Git también es capaz de detectar conflictos, es decir, si el trabajo de dos personas se solapa, nos advierte sobre el conflicto a la hora de fusionar las ramas.

**Github**: Github es por un lado un gestor de proyectos y un repositorio online de control de versiones y por otro lado una plataforma social de trabajos en grupo. Para depositar o descargar proyectos a este o desde este repositorio, se utiliza git. Github es una apliaciónde Microsoft.

**Algoritmo**: Conjunto de instrucciones bien definidas y estructuradas para realizar una tarea determinada. Un algoritmo debe ser preciso, debe estar perfectamente definido, debe ser finito, debe legible y debe constar de tres pasos: entrada, proceso y salida.

**Diagrama de flujo:** Es una representación gráfica de los pasos de un algoritmo**.** En él podemos representar mediante símbolos los pasos del algoritmo y mediante flechas marcar el flujo entre estos pasos, desde el inicio hasta el final.

**Ada Lovelace**: Fue una pionera de la informática (Londres, 1815-1852), publicó el primer algoritmo informático. Creó este algoritmo para la máquina creada por el **matemático inglés Charles Babbage**, esta “máquina analítica” era capaz de calcular secuencias de números. Este código nunca fue probado, porque la maquina nunca se construyó. Introdujo también los conceptos de bucle (grupo de instrucciones que se ejecutan varias veces) o subrutina (parte de programa que puede ser llamado en cualquier momento).

**Algoritmos de cifrado simétrico:** la criptografía simétrica también conocida como Criptografía de una sola clave o Criptografía de Clave secreta se basa en que el emisor del mensaje y el receptor del mensaje comparten una clave secreta. El mensaje se cifra y se descifra con la misma clave. Por tanto, la seguridad de este tipo de cifrado recae únicamente en la clave y no en el algoritmo de cifrado y descifrado. Es importante que la clave sea muy difícil de adivinar.

**Algoritmo de cifrado asimétrico:** En este caso existen dos claves diferentes una clave pública que puede conocer cualquiera, con la que se cifra el mensaje y una clave privada, para descifrarlo, que sólo conoce el receptor del mensaje. Por tanto, no hay envío de claves y en este sentido, es más seguro que el cifrado simétrico. En contrapartida, consume más recursos y es más lento.

**Algoritmo de cifrado híbrido**: Este tipo de cifrado, es una mezcla de los anteriores, en un intento por aunar las ventajas de cada uno, pretende ser seguro a la vez que no tan lento. Se generan para el receptor la clave pública y la privada. Se cifra el mensaje, el receptor manda su clave pública con la que se cifra la clave que se ha usado para cifrar el mensaje. Se envía el archivo y la clave del menaje cifrada.