**DICCIONARIO**

# Nivel 1 (Turing)

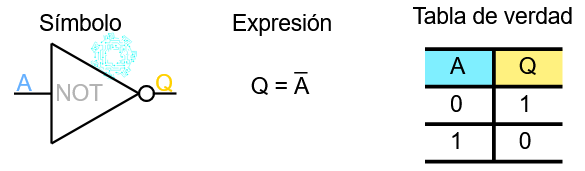
## Tarea 5 – ¿Qué es la IA?

* Orígenes de la IA
  + 1950, Alan Turing
    - Publica un primer artículo con el titulo “Computing machinery and intelligence”
    - Test de Turing
      * Para saber si una máquina es “inteligente” o no (sigue vigente)
  + 1956, John McCarty, Marvin Misky y Claude Shannon
    - Conferencia Darmouth “La ciencia e ingenio de hacer máquinas inteligentes, especialmente programas de cálculo inteligente”.
  + 1997
    - El supercomputador Deep Blue de IBM gana al campeón mundial de ajedrez Gari Kasparov.
* Definición de Inteligencia Artificial
  + 1959, Arthur Samuel
    - Campo de estudio que da a los ordenadores la capacidad de aprender sin ser programados de manera explícita.
  + 1997, Tom Mitchell
    - Se dice de un programa de ordenador que aprende de la experiencia E, con respecto a una tarea T y una medida de rendimiento R, si su rendimiento T, medido por R, mejora con la experiencia E.
  + Aurélien Geron
    - Ciencia (y el arte) de programar ordenadores para que aprendan a partir de datos.
  + The Egg
    - La IA se puede definir como cualquier sistema que es capaz de cambiar su estado de forma autónoma basado en unos datos de entrada al mismo. Es por ello que se dice que los datos son el combustible de la IA.
* Algunas de las tecnologías dentro de la IA
  + Machine Learning, Deep Learning, Computer Vision o Natural Language Processing.

## Tarea 12 – Los transistores

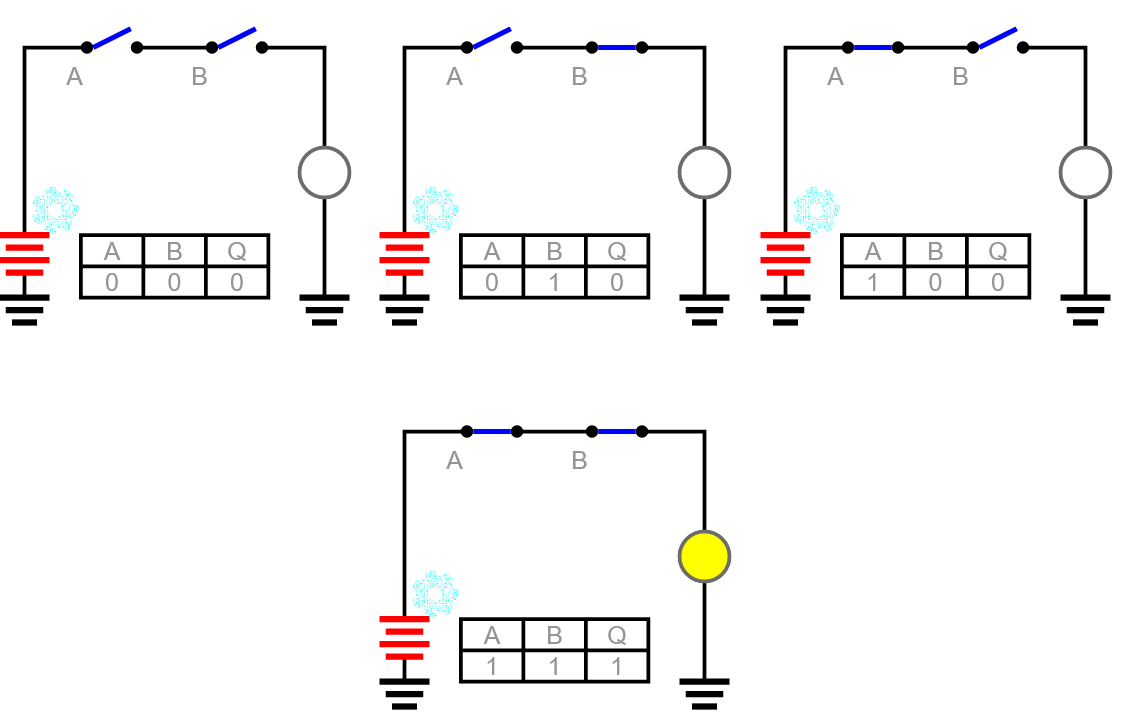
TRANSISTORES

* Tipos
  + Compuerta NOT/INV
    - Su propósito es producir una salida inversa o contraria a su entrada
    - 1: activo
    - 0: inactivo

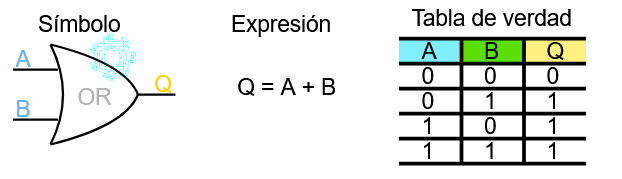


* + Compuerta AND
    - Todo o nada
    - La expresión del medio de la foto es Álgebra de Boole
    - Para tener una salida activa es necesario que sus entradas tengan un estado binario 1

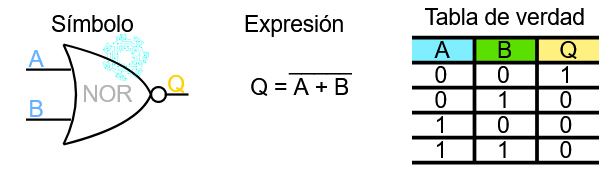




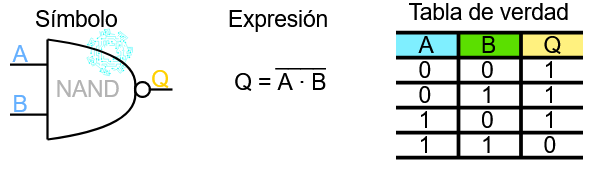
* + Compuerta OR
    - Cualquiera o todo
    - Esta compuerta se encuentra en estado activo siempre y cuando una de sus entradas tenga un estado binario activo “1”



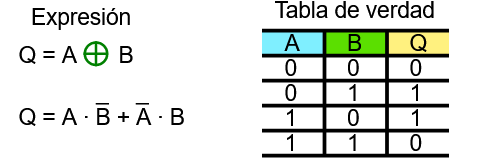
* + Compuerta NOR
    - Versión inversa de la compuesta OR

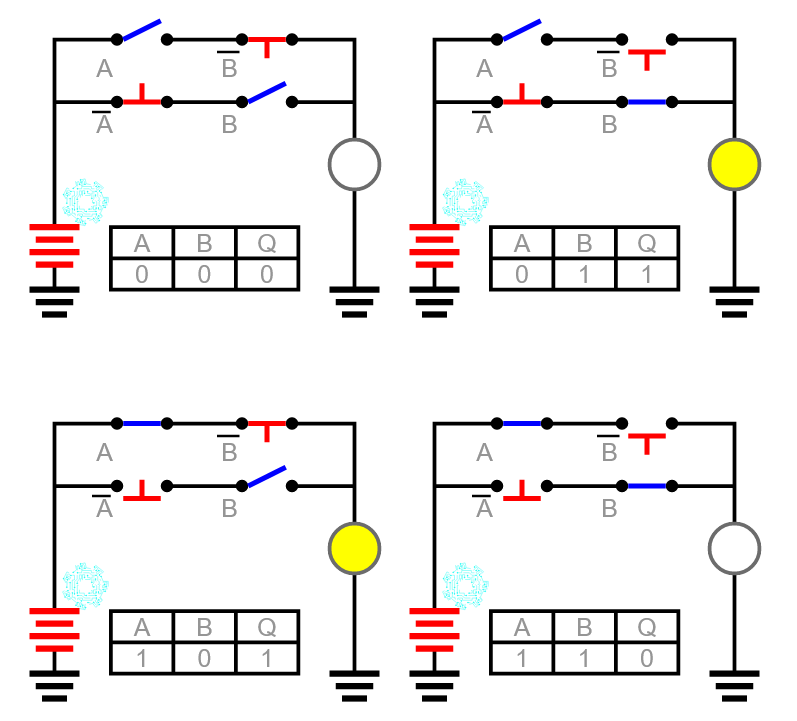


* + Compuerta NAND
    - Versión inversa de NAND

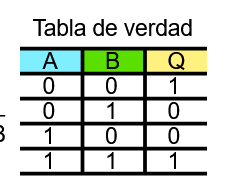


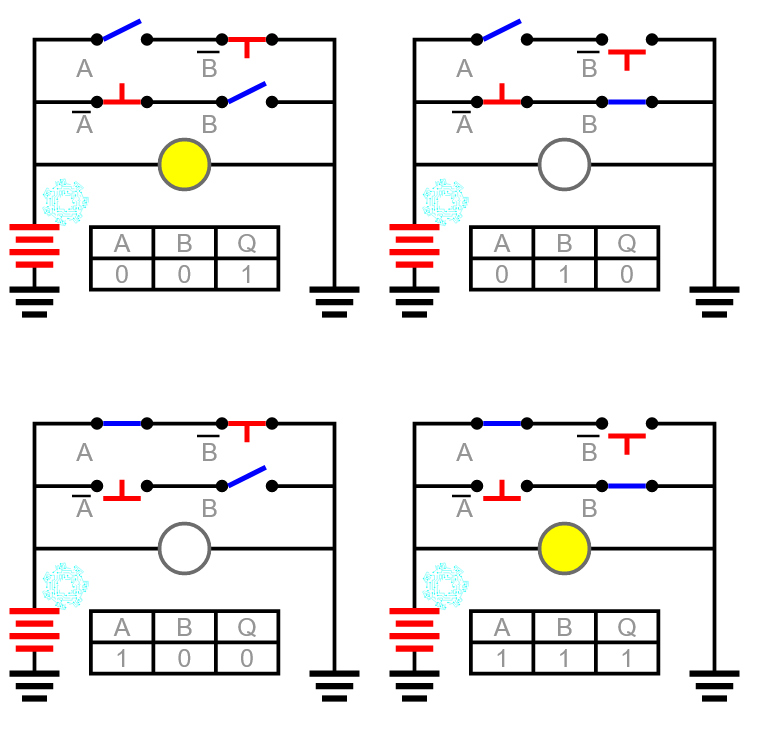
* + Compuesta XOR
    - Algunos pero no todos
    - La salida tiene un estado activo “1” al tener las entradas en estados diferentes





* + XNOR
    - Negación de la compuerta XOR
    - Cuando las entradas sean iguales se representará una salida en estado “1” y si son diferentes la salida será un estado “0”

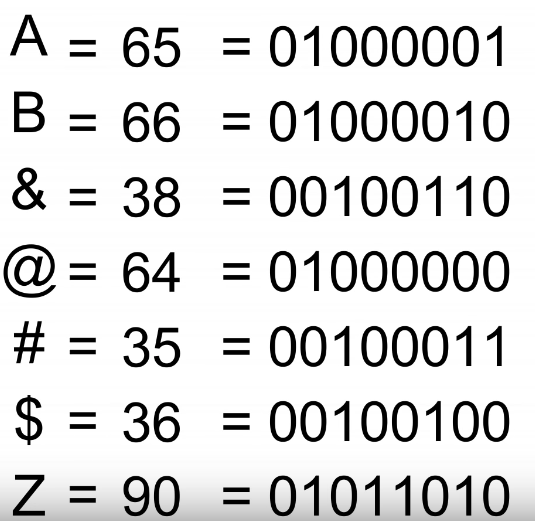


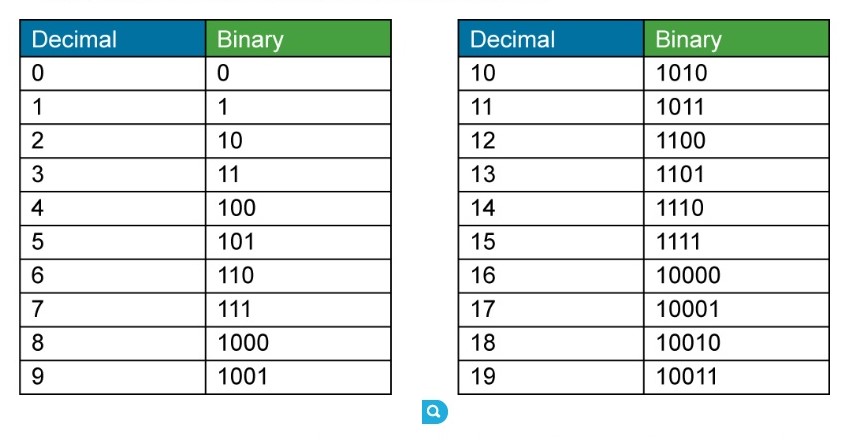


* CALCULADORA
  + Números binarios
    - Las posiciones son potencias de 2.

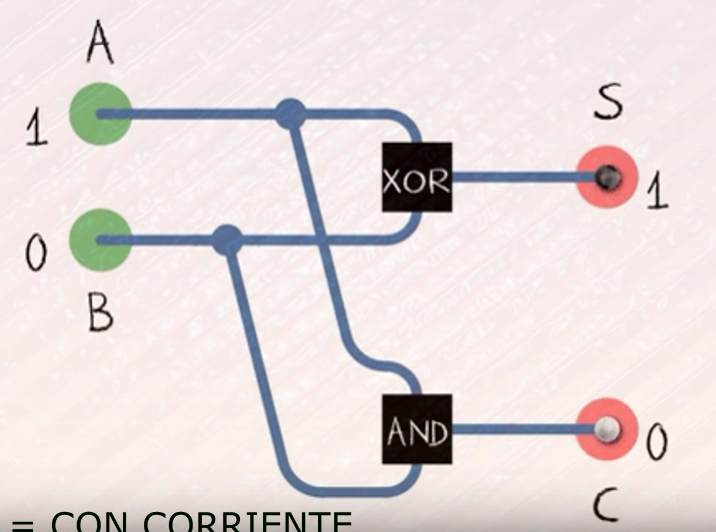


* + - Las letras y símbolos tienen asociados unos números para que un ordenador pueda entender nuestro idioma

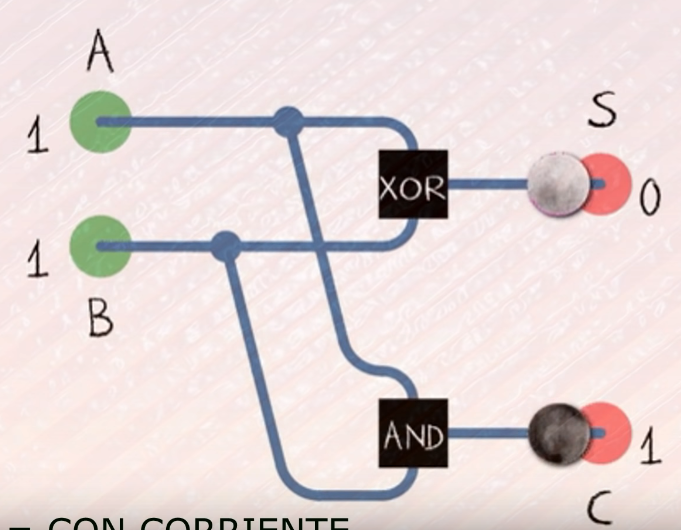




* + 1+0



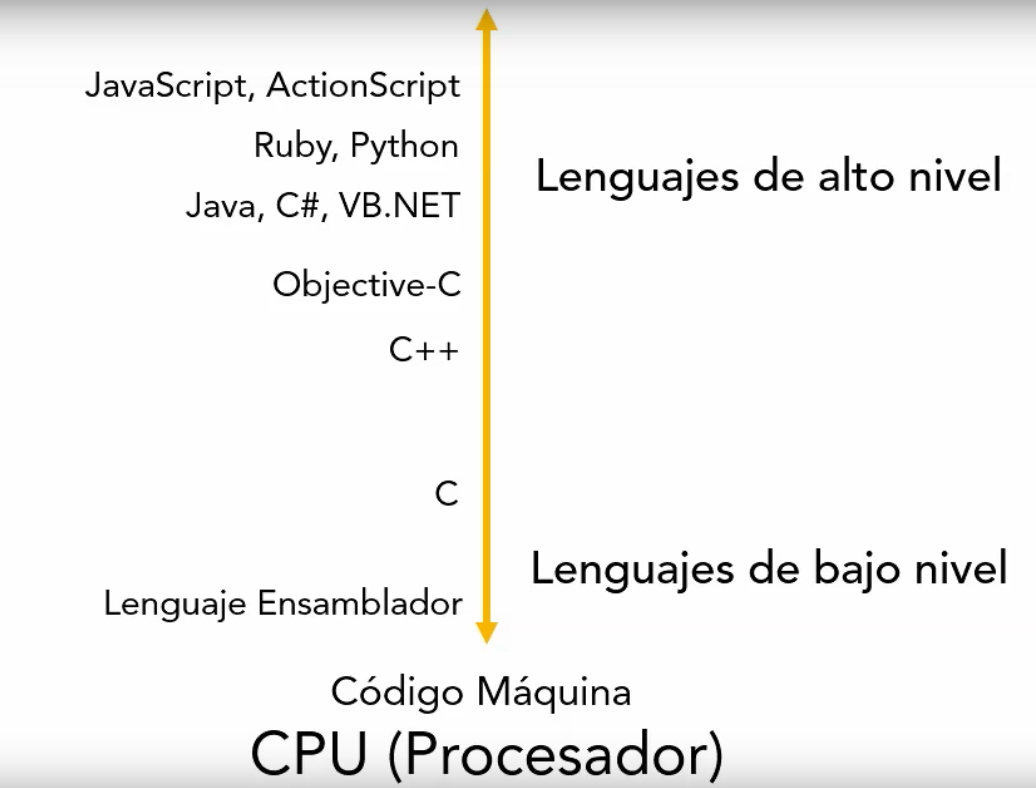
* + 1+1



* Bit y Byte
  + Bit
    - Binary digit (digito binario)
    - Un bit tiene solamente un valor= 0 o 1
    - Varios bits combinados entre sí dan origen a otras unidades: “byte”, “mega”, “giga”, “tera”
  + Byte
    - Es un conjunto de 8 bits
    - Se puede hacer una correspondencia biunívoca entre cada número decimal (0 a 9), las letras mayúsculas y minúsculas (A hasta Z), los símbolos matemáticos, la puntuación, y demás símbolos, con un respectivo byte.
* CPU 32 bits/64 bits
  + Los sistemas de 32 bits almacenan sus datos en piezas de 32 bits
  + Los sistemas de 64 bits almacenan sus datos en piezas de 64 bits
  + Al trabajar con piezas más grandes, puede hacer más en menos tiempo

## Tarea 14 – ¿Qué es el lenguaje de programación?

* Un lenguaje de programación lo componen una lista de instrucciones que organizadas de una manera conforman programas
  + Alto nivel
    - Se caracteriza por expresar los algoritmos de una manera adecuada a la capacidad cognitiva humana, en lugar de la capacidad con que los ejecutan las máquinas.
    - Estos lenguajes permiten una máxima flexibilidad al programador a la hora de abstraerse o de ser literal
    - Tendrán que ser compilados o interpretados para transformar el código fuente en código de maquina (código binario)
    - Ej.: Python, R, C++…
  + Bajo nivel
    - Es aquel en el que sus instrucciones ejercen un control directo sobre el hardware y están condicionados por la estructura física de las computadoras que lo soportan
    - Mínima abstracción entre el lenguaje y el hardware
    - Dos tipos:
      * Lenguaje ensamblador
        + Es el primer intento de sustitución del lenguaje de máquina por uno más cercano al utilizado por los humanos
        + Un programa escrito en éste lenguaje es almacenado como texto
        + Ej: MOV AL, 61h (asigna el valor hexadecimal 61 al registro “AL”)
      * Lenguaje máquina
        + Es el más primitivo de los lenguajes y es una colección de dígitos binarios o bits (0 y 1) que la computadora lee e interpreta.
        + Ejemplo: 10110000 01100001



## Tarea 15 – ¿Qué es el lenguaje de programación?

* Frontend
  + Frontend es la parte de un programa o dispositivo a la que un usuario puede acceder directamente.
  + Son todas las tecnologías de diseño y desarrollo web que corren en el navegador y que se encargan de la interactividad con los usuarios
  + HTML, CSS Y Javascript
* Backend
  + Backend es la capa de acceso a datos de un software o cualquier dispositivo, que no es directamente accesible por los usuarios.
  + Además contiene la lógica de la aplicación que maneja dichos datos.
  + También es el que accede al servidor, que es una aplicación especializada que entiende la forma de como el navegador solicita cosas.
  + Python, PHP, Ruby, C#, Java
* Librería
  + Una librería es un programa que resuelve algo, desarrollado generalmente por terceros, que se pueden reutilizar para no tener que hacer todo desde el principio.

## Tarea 22 – El algoritmo del lechero

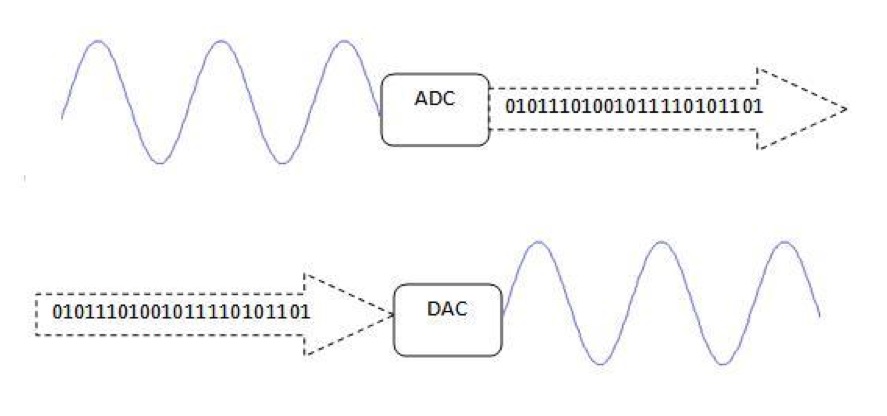
* Programación dinámica

## Tarea 23 – Cifrado y descifrado con el solitario

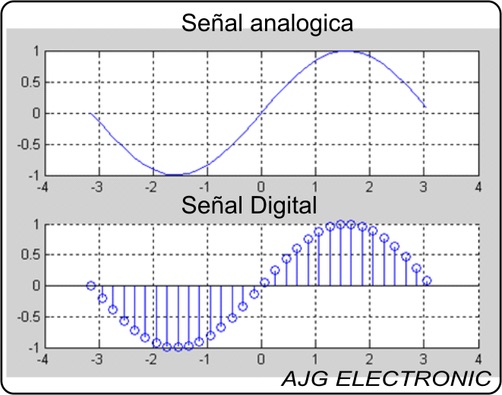
* Cifrado
* Descifrado

## Tarea 24 – Construye un simulador

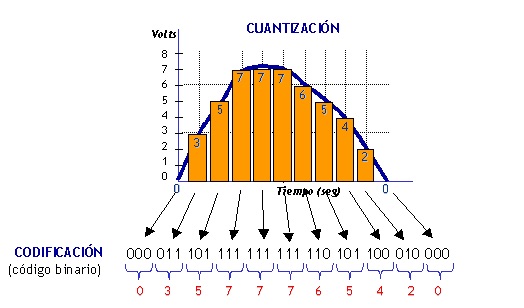
* Digital
  + Un sistema digital es cualquier sistema que permita crear, decodificar, transmitir o guardar información que se encuentra representada en cantidades tan restringidas que sus señales de entrada y salida solo admiten valores discretos.
    - Una variable discreta es una [variable](https://es.wikipedia.org/wiki/Variable_estad%C3%ADstica) que no puede tomar algunos valores dentro de un mínimo conjunto numerable, quiere decir, no acepta cualquier valor, únicamente aquellos que pertenecen al conjunto. Como ejemplo, el número de animales en una granja (0, 1, 2, 3, 4, 5, 6, 7, …). Otro ejemplo sería el número de hijos en una familia (1; 2; 3; 4; …).
    - Una variable continua toma valores a lo largo de un continuo, esto es, en todo un [intervalo](https://es.wikipedia.org/wiki/Intervalo_(matem%C3%A1tica)) de valores. Como ejemplo, la estatura de una persona (1,72 m; 1,719 m; 1,718 6 m....).
  + Ventajas de los sistemas digitales
    - Menor tamaño: los dispositivos basados en sistemas digitales tienden a hacerse cada vez más pequeños.
    - Eficiencia: la información digital se almacena rápidamente y en cantidades cada vez más grandes.
    - Precisión: como los sistemas digitales solo admiten valores discretos, son mucho más precisos.
    - Diseño: los dispositivos basados en sistemas digitales tienden a hacerse cada vez más fáciles de diseñar
    - Estabilidad: los sistemas digitales son menos susceptibles al ruido, es decir, a todas las posibles perturbaciones de la señal.
  + Desventajas de los sistemas digitales
    - Conversión: las naturaleza de las variables físicas es analógica (sonido, temperatura, distancia, peso) por lo tanto, es necesario usar un conversor para transformarlas en datos digitales.
    - Ancho de banda: la transmisión de señales en un sistema digital requiere de un ancho de banda mucho mayor que un sistema analógico.
    - Alteración: los sistemas digitales pueden alterarse o manipularse con relativa facilidad con respecto a los analógicos.
* Analógico
  + **Sistema analógico** es cualquier sistema cuyas señales se representan con valores continuos, es decir, que admite números o valores infinitos.
  + Los datos que forman parte de la naturaleza son de origen analógico: la temperatura, la distancia, el sonido, voltaje, imágenes, etc. Si bien todas estas variables se pueden convertir a datos digitales son, originalmente, analógicas.
  + Una señal analógica es un voltaje o corriente que varía suave y continuamente. Una onda senoidal es una señal analógica de una sola frecuencia. Los voltajes de la voz y del video son señales analógicas que varían de acuerdo con el sonido o variaciones de la luz que corresponden a la información que se está transmitiendo.



* + El conversor ADC (Analog-to-Digital Converter – Conversor Analógico Digital) tiene que efectuar los siguientes procesos:
    - Muestreo de la señal analógica
      * Para convertir una señal analógica en digital, el primer paso consiste en realizar un muestreo (sampling) de ésta, o lo que es igual, tomar diferentes muestras de tensiones o voltajes en diferentes puntos de la onda senoidal. La frecuencia a la que se realiza el muestreo se denomina razón, tasa o también frecuencia de muestreo y se mide en kilohertz (kHz). En el caso de una grabación digital de audio, a mayor cantidad de muestras tomadas, mayor calidad y fidelidad tendrá la señal digital resultante.



* + - Cuantización de la señal analógica
      * La cuantización representa el componente de muestreo de las variaciones de valores de tensiones o voltajes tomados en diferentes puntos de la onda sinusoidal, que permite medirlos y asignarles sus correspondientes valores en el sistema numérico decimal, antes de convertir esos valores en sistema numérico binario.



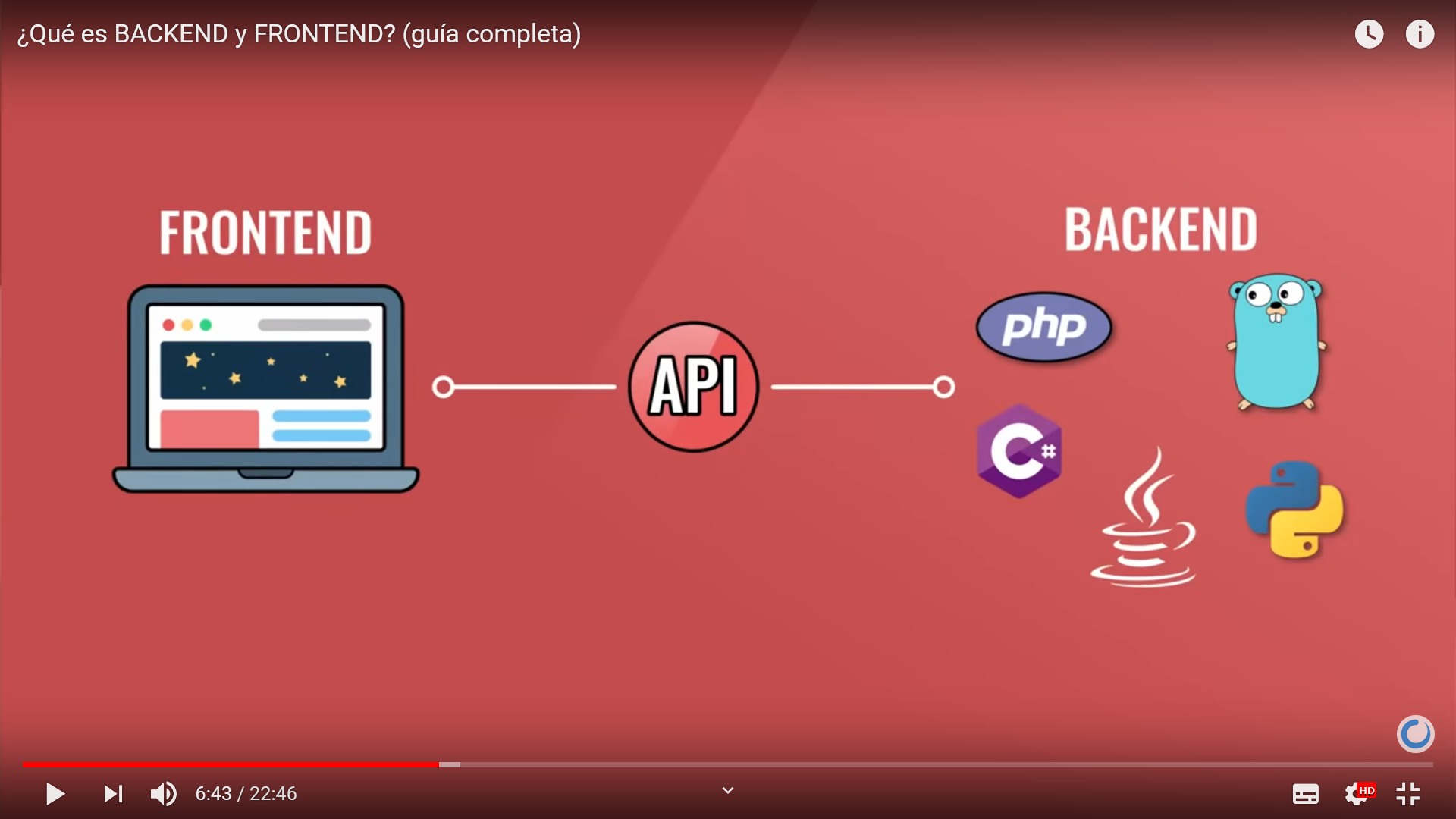
* + - Codificación de la señal en código binario
      * Después de realizada la cuantización, los valores de las tomas de voltajes se representan numéricamente por medio de códigos y estándares previamente establecidos. Lo más común es codificar la señal digital en código numérico binario.

## Tarea 25 – Paradigma cliente-servidor

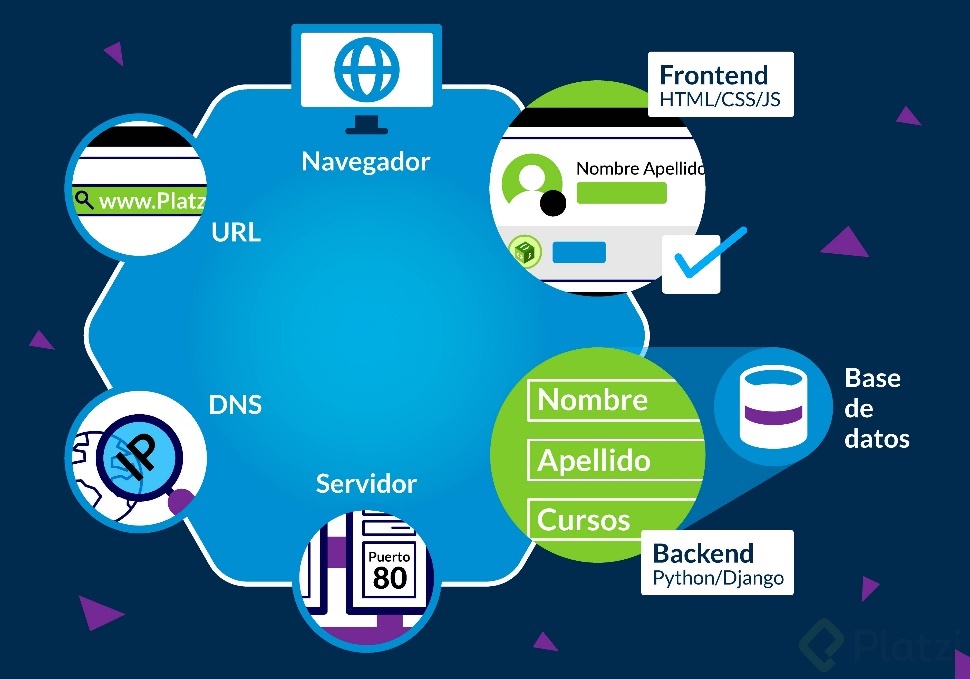
* Arquitectura cliente-servidor (son la clave de internet)



* + Como funciona el proceso desde que entras en Google hasta que aparece en el navegador
    - Cuando hacemos una abrimos una página web nosotros escribimos la dirección, como por ejemplo google.com. Sin embargo, para poder llegar a google.com se necesita saber la IP de Google, que no es otra cosa que una dirección numérica única que identifica de forma única al servidor.
    - Para poder obtener la IP se hace una consulta a un servidor DNS. Esto el usuario no lo ve porque sucede internamente en nuestro sistema.
    - Una vez que nuestro sistema operativo resolvió la **consulta DNS** y sabe cómo encontrar el servidor, nuestro sistema operativo a través del **protocolo TCP/IP** negocia una comunicación con el servidor remoto que aloja la página que queremos ver.
    - Una vez establecida la comunicación, luego mediante el **protocolo HTTPS** y usando una petición tipo de petición GET se solicita la URL en cuestión.
    - En este paso el servidor web debe determinar primero **a que sitio corresponde la petición** (ya que en un mismo servidor pueden alojar varios dominios) y lo hace a través del header recibido observando la URL que se solicitó.
    - Gracias al archivo .htaccess puedes crear URL amigables para ayudar al SEO o simplemente hacer las URL más sencillas, se pueden establecer políticas de seguridad, permisos de acceso o denegación a archivos, filtrados de IP, modificar las cabeceras de respuesta u otros valores de configuración, las opciones son sumamente diversas.
    - A su vez en este punto el servidor debe determinar que se está solicitando, por ejemplo si se trata de una imagen o un archivo HTML el servidor lo leerá del disco SSD y lo enviará al cliente, en cambio si se trata de una página .php o .asp el servidor primero ejecutará el código de dicho script para obtener el contenido que devolverá, que puede ser HTML, un JSON, texto o incluso una imagen o un archivo, a esto es lo que se llama contenido dinámico debido a que no existe en el servidor como tal sino que es generado en el momento para ser mostrado.
    - Una vez que el servidor obtuvo el contenido o datos devolverá una cabecera de respuesta donde por ejemplo especifica el tipo de datos devuelto, en ella indicará si es HTML, un JSON, una imagen o archivo de audio, entre otros, esto facilita al navegador saber que tiene que hacer con ese fichero y cómo debe tratarlo posteriormente.
    - Cada respuesta del servidor va acompañada de un número de identificación que indica que ocurrió. Por ejemplo, si en una petición GET obtenemos el código 200 quiere decir que esta todo correcto y sin errores. Si por el contrario responde con un error 500 será un error interno del servidor, y si recibimos un error 404 es porque la URL no se encontró.
  + Servidor vs Servidor Web
    - Servidor es la máquina física
    - Un servidor web es en realidad un servicio web, es decir, un software que se ejecuta en ese servidor físico o virtual y que despacha tráfico por el protocolo HTTP. En resumen, es un programa informático que se encarga de recibir solicitudes de clientes que solicitan páginas HTML u otro tipo de archivos como imágenes y entregarlos al usuario.
    - Estas solicitudes pueden devolver contenido de dos tipos: **contenido estático** o contenido dinámico.
      * El contenido estático es el contenido que es entregado sin procesar en alguna forma, es decir tal como está almacenado se envía al usuario que lo solicito, como por ejemplo una imagen.
      * El otro tipo de contenido es el llamado **dinámico**, esto significa que el servidor devuelve contenido que no existe como tal en el servidor, sino que es generado en el momento, como por ejemplo la página de resultados de un buscador.
* Frontend y Backend



* + Frontend
    - Es la parte de un programa o dispositivo a la que un usuario puede acceder directamente. Son todas las tecnologías de diseño y desarrollo web que corren en el navegador y que se encargan de la interactividad con los usuarios.
    - Un programador Frontend debe saber de códigos [HTML, CSS y JavaScript]([HTML, CSS y JavaScript](https://platzi.com/cursos/html5-css3/)) para poder usar algunos frameworks o librerías que expanden sus capacidades para crear cualquier tipo de interfaces de usuarios. [React](https://platzi.com/cursos/react/), [Redux](https://platzi.com/cursos/redux/), [Angular](https://platzi.com/cursos/angular/), [Bootstrap, Foundation](https://platzi.com/cursos/fw-frontend/), [LESS](https://platzi.com/cursos/less/), [Sass](https://platzi.com/cursos/sass/), [Stylus](https://platzi.com/cursos/stylus/) y [PostCSS](https://platzi.com/cursos/postcss/) son algunos de ellos.
  + Backend
    - Backend es la capa de acceso a datos de un software o cualquier dispositivo, que no es directamente accesible por los usuarios, además contiene la lógica de la aplicación que maneja dichos datos. El Backend también accede al servidor, que es una aplicación especializada que entiende la forma como el navegador solicita cosas.
    - Algunos de los lenguajes de programación de Backend son [Python](https://platzi.com/cursos/python/), [PHP](https://platzi.com/cursos/php/), [Ruby](https://platzi.com/cursos/ruby/), [C#](https://platzi.com/cursos/c-sharp/) y [Java](https://platzi.com/desarrollo-java/).



## Tarea 31 – El proceso ETL

* Proceso ETL (Extract, Transform, Load)
  + Proceso clave para pasar de formatos desestructurados como JSON o XML, con valores nulos, datos errados, caracteres inválidos, registros duplicados y demás problemas a datos totalmente listos para ser analizados.
  + Este proceso tiene tres pasos:
    - Extract
      * Traer los datos del origen
      * El origen de datos o “source” puede contener múltiples fuentes de datos.
    - Transform
      * Transformación de los datos para modelar según lo queramos. Operaciones típicas:
        + Filtrar
        + Eliminar duplicados
        + Calcular datos nuevos
        + …
      * Todo esto se realiza en el área de “staging”, un repositorio temporal
    - Load
      * Una vez los datos están transformados y listos en el área de “staging”, se procede a cargarlos en la base de datos de analítica. Esta base de datos comúnmente es un datawarehouse en donde conviven distintos repositorios de datos.

## Tarea 34 – ¿Y qué es una API?

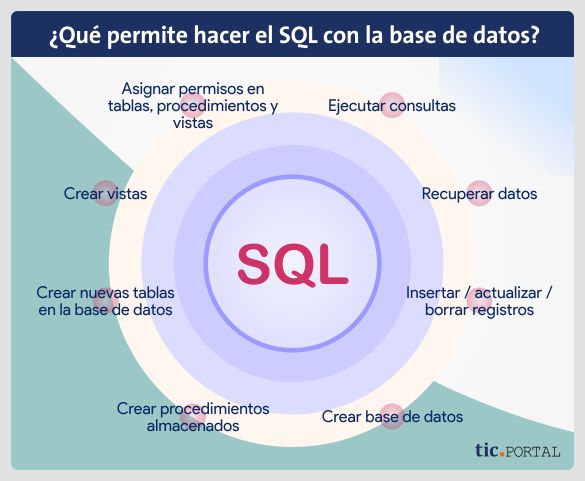
* ¿Qué es una API?
  + Una API es un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones. En otras palabras, es una especificación formal sobre cómo un módulo de un software se comunica o interactúa con otro.
  + Las API permiten que sus productos y servicios se comuniquen con otros, sin necesidad de saber cómo están implementados.
  + Por ejemplo, cuando el usuario compra entradas a través de la página web de una sala de cine e introduce la información de su tarjeta de crédito, la web usa una API para enviar dicha información de forma remota a otro programa que verifica si los datos bancarios son correctos.
  + Otro ejemplo: imagina una empresa distribuidora de libros. Podría ofrecer a los clientes una aplicación que les permita a los empleados de la librería verificar la disponibilidad de los libros con el distribuidor. El desarrollo de esta aplicación podría ser costoso, estar limitado por la plataforma y requerir mucho tiempo de desarrollo y mantenimiento continuo. Otra opción es que la distribuidora de libros proporciona una API para verificar la disponibilidad en inventario. Existen varios beneficios de este enfoque:
    - Permite que los clientes accedan a los datos con una API que les ayude a añadir información sobre su inventario en un solo lugar.
    - La distribuidora de libros podría realizar cambios en sus sistemas internos sin afectar a los clientes, siempre y cuando el comportamiento de la API fuera el mismo.
    - Con una API disponible de forma pública, los desarrolladores que trabajan para la distribuidora de libros, los vendedores o los terceros podrían desarrollar una aplicación para ayudar a los clientes a encontrar los libros que necesiten. Esto podría dar como resultado mayores ventas u otras oportunidades comerciales.
  + Además, es independiente del lenguaje de programación que se utilice en la aplicación ya que se utiliza un protocolo de comunicación común y estándar (JSON, XML, ...)

## Tarea 36 – Introducción a las bases de datos

* Una base de datos es una herramienta que **recopila datos, los organiza y los relaciona** para que se pueda hacer una rápida búsqueda.
* Generalmente los datos están ordenados en tablas.
* El **sistema de gestión de base de datos** (DBMS, Data Base Management System) es un software que sirve para crear y acceder a los datos de la base de datos, como, por ejemplo, SAP HANA o Amazon RDS.
* Hay varios tipos:
  + Base de datos relacional
    - La **base de datos relacional** es una recopilación de la información empresarial organizada de tal forma que se puede consultar, actualizar, analizar y sacar los datos fácilmente. La información se encuentra en **tablas y campos** relacionados entre sí.



* + - El lenguaje estándar para el acceso y manipulación de la base de datos es **SQL** (Structured Query Language).

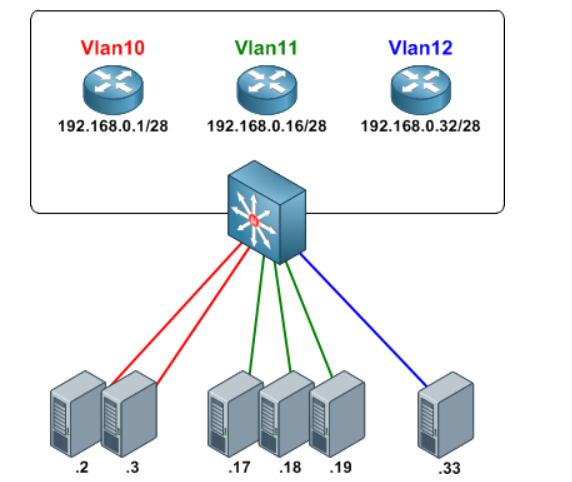


* + Base de datos distribuida
    - Consiste en el almacenamiento de porciones de la base de datos en diferentes ubicaciones físicas y, por tanto, el procesamiento está distribuido o replicado entre los distintos puntos de una red de trabajo.
    - La base de datos distribuida tiene una **mayor disponibilidad de los datos** debido a sus múltiples ubicaciones. De esta forma, si una de las bases de datos fallase, se podría seguir funcionando correctamente, aunque puede que un poco ralentizado.
    - Sin embargo, esta misma característica de estar distribuido puede provocar que haya **duplicidad de los datos y un menor nivel de seguridad.**
    - Dos tipos:
      * Homogénea: Las distintas ubicaciones están conectadas y tienen el mismo DBMS
      * Heterogénea: son independientes entre sí.
  + Base de datos NoSQL (Not Only SQL)
    - Esto se debe a que este tipo de base de datos **suele evitar el uso del SQL** o lo usa de apoyo, pero no como consulta.
    - El hecho de evitar el SQL es porque se usa para proyectos en los que se necesita trabajar en la base de datos con un gran volumen. En las bases de datos con lenguaje SQL, los distintos atributos de un elemento, están en diferentes columnas, mientras que en una NoSQL todos los atributos se encuentran en una misma columna, ahorrando espacio.
    - Ventajas
      * Alto rendimiento y baja exigencia: puede ejecutarse en máquinas con recursos modestos.
      * Escalabilidad horizontal: se pueden añadir nodos de forma horizontal para subir el rendimiento.
      * Grandes volúmenes de datos
      * Flexibilidad
    - Desventajas
      * Funciones de Fiabilidad: Cómo no admite funciones de fiabilidad lleva a los desarrolladores a implementar su propio código, agregando mayor complejidad a los sistemas.
      * Aplicabilidad: la falta de inclusión de funciones de fiabilidad limita la aplicabilidad de estas BBDD a sectores delicados como el bancario.
      * Incompatibilidad en consultas SQL
    - Ejemplos de lenguajes usados en NoSQL: JSON, CQL, GQL
  + Base de datos orientada a objetos
    - Representa los datos en forma de objetos y clases. El objeto puede ser desde un resultado de búsqueda a una tabla; y una clase es una colección de objetos.
    - Este tipo de base de datos están generalmente están escritas en **lenguajes de programación orientados a objetos** como Java, C o Smalltalk.
  + Base de datos gráfica
    - Se diferencian de las anteriores en que están especializadas en **establecer relaciones entre los datos de forma visual y navegar por dichas relaciones**. Para leer la información, hay que leer los nodos o conectores (puntos de conexión de los datos de las tablas), generando un lenguaje natural.
  + Base de datos autogestionadas
    - Las bases de datos de autogestionadas mantienen la potencia y las ventajas del modelo relacional, pero utilizan la IA, el aprendizaje automático y la automatización para supervisar y mejorar el rendimiento de las consultas y las tareas de administración.

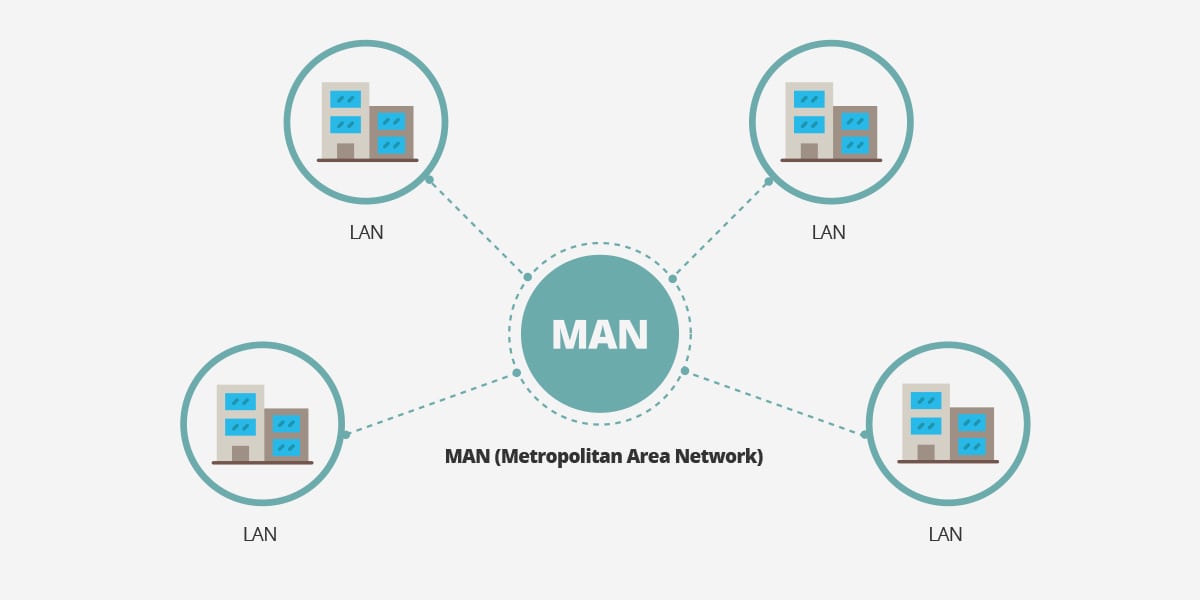
## Tarea 37 – Empezando con las de comunicación / Tarea 39 – Simulador de redes

**Redes**

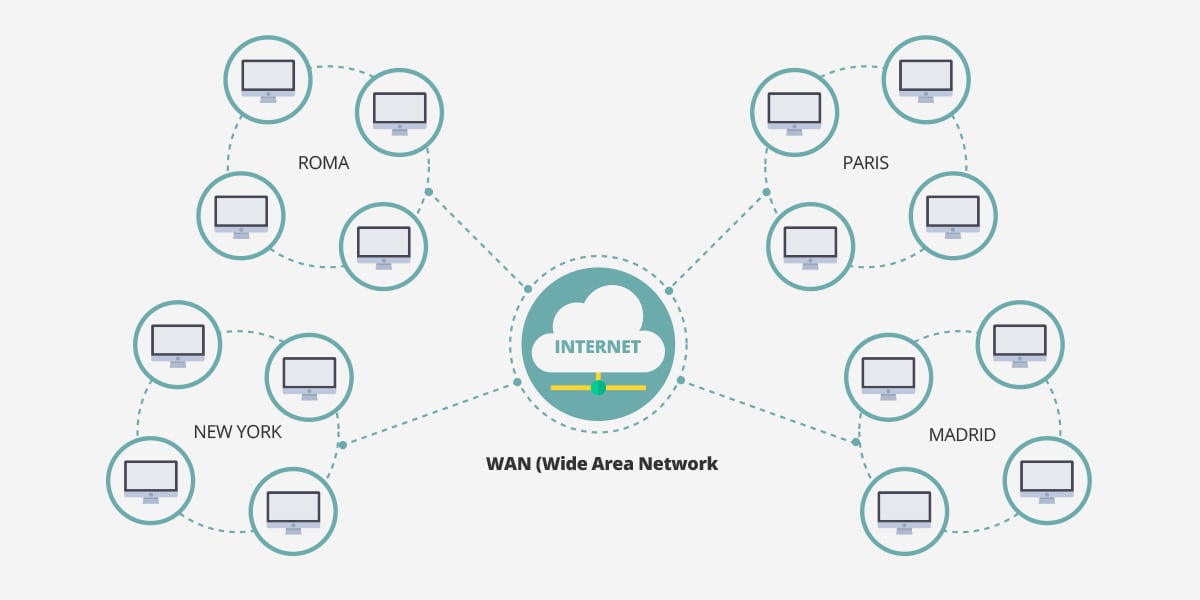
* Una red es la unión de dos o más ordenadores de manera que sean capaces de compartir recursos, ficheros, directorios… Para crear la red es necesario un hardware que una los dispositivos (tarjetas, cables) y un software que implemente las reglas de comunicación entre ellos (protocolos y servicios).
* Tipos de red:
  + LAN (Local Area Network)
    - Son redes de pequeña extensión.
    - La mayoría de las veces, **una LAN se limita a una solo piso, edificio o grupo de edificios** y puede servir para conectar dos o tres usuarios (por ejemplo, en una red de oficina pequeña) o para varios cientos de usuarios en una oficina más grande.
    - Hay dos tipos:
      * WLAN
        + Comunicación inalámbrica.
        + Las redes WIFI por ejemplo son una red WLAN.
      * LAN
        + Con el cable ethernet.
    - VLAN
      * Del inglés Virtual LAN (Red de área local y virtual), es un método que permite crear redes que lógicamente son independientes, aunque estas se encuentren dentro de una misma red física. De esta forma, un usuario podría disponer de varias VLANs dentro de un mismo router o switch.
      * En lo que concierne a la seguridad, hay que tener en cuenta que los dispositivos pertenecientes a una VLAN no tienen acceso a los que se encuentren en otras y viceversa. Resulta útil cuando queremos segmentar los equipos y limitar el acceso entre ellos por temas de seguridad.
      * Enlace Trunk
        + Es un enlace que se configura en uno o más puertos de un **switch** para permitir el paso del tráfico de las distintas VLANs que hemos configurado. Este enlace puede funcionar en una conexión de switch a otro switch o bien, de un switch a un router, e incluso de un switch a un servidor que soporte el estándar 802.1Q para «pasarle» varias VLANs simultáneamente.
        + El protocolo esencial que da vida al enlace troncal es el que pertenece al estándar **IEEE 802.1Q**. Esto permite que las tramas Ethernet viajen a través de la red con una «etiqueta» que contiene el identificador de la VLAN.
        + Por ejemplo, si tienes una misma VLAN (mismo código y nombre) en dos switches distintos, el enlace trunk permite que la información fluya entre las dos partes.



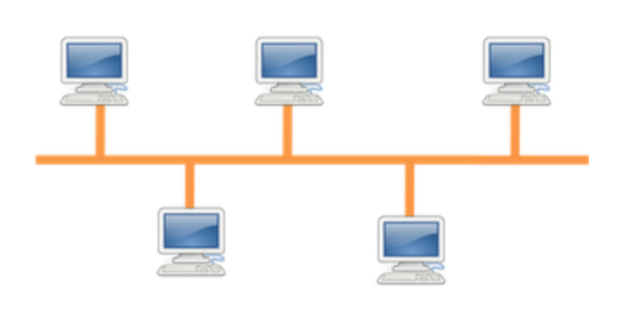
* + - * VLAN trunking
        + Es la práctica de tener tramas de Ethernet de dos o más Vlan que fluye a través de un solo puerto y el cable, lo que permite al router hablar con dos o más Vlan sin necesidad de un puerto físico de cada uno de ellos.
        + DOT1Q (nombre real 802.1 1) es un protocolo para la transmisión de información de VLAN junto con las tramas de Ethernet.
        + Esta “etiqueta” dice lo siguiente: ”este fotograma pertenece a la VLAN X”
  + MAN (Metropolitan Area Network)
    - Una red de área metropolitana (MAN por las siglas en inglés de Metropolitan Area Network) consiste en computadoras compartiendo recursos entre sí en áreas de cobertura de mayor tamaño que una LAN, pero menor que una WAN. Funcionan de forma muy parecida a una red de área local, pero cumplen estándares tecnológicos diferentes. Estas mejoras son necesarias para subsanar los problemas de latencia (retardo en la entrega de información) y pérdida de calidad de la señal en interconexiones que abarcan largas distancias.



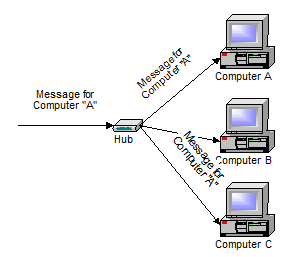
* + WAN (Wide Area Network)
    - Las redes de área amplia (WAN por las siglas de Wide Area Network), son redes informáticas LAN y MAN interconectadas entre sí. Sus nodos están separados por distancias que pueden abarcar continentes enteros. Los integrantes de esas redes no necesariamente están conectados físicamente. Hacen uso de servicios de microondas y satelitales para integrar sus diferentes nodos.
    - Las conexiones 3G/4G/5G de los teléfonos móviles se basan en este tipo de redes (WWAN).



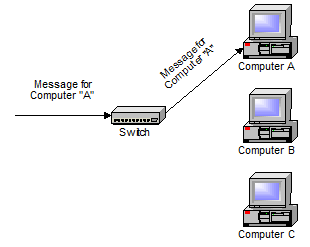
* Topología de redes
  + En anillo
    - Consiste en conectar las estaciones una en serie con la otra formando un anillo cerrado. La información debe pasar de una estación a otra hasta que llega al destinatario de la misma, generalmente la información es de tipo unidireccional.
    - Un inconveniente de esta topología es que si una estación se avería la red deja de funcionar adecuadamente.
  + En bus
    - Consiste en conectar todas las estaciones a un bus común.



* + - Principales características:
      * La información circula en ambas direcciones.
      * No hay ningún nodo central que controle la red.
      * La información se transmite por todo el bus. Por ello, todos los equipos reciben todas las señales.
      * Es sencillo añadir nuevos receptores.
      * La rotura del cable principal (bus) provoca la caída del servicio a todos los dispositivos de la red.
  + En estrella
    - Esta estructura es la más común y la que más flexibilidad proporciona a la hora de realizar modificaciones y ampliaciones. En ella cada uno de los nodos está conectado mediante un enlace directo a un centro de comunicaciones, concentrador "**hub**" o nodo central.
    - En la actualidad es el sistema más extendido.
    - Facilita la supervisión y control de la información al obligar a pasar los mensajes por el concentrador.
    - La fiabilidad de este tipo de red se ve aumentada pues la puesta fuera de servicio de uno de los terminales no interfiere en el funcionamiento del resto de la red.
    - El nodo central es el cuello de botella.
    - Existen distintos tipos de nodos centrales:
      * Hub
        + **Tiene la función de interconectar los ordenadores de una red local**. Comparado con el switch y el router, es mucho más simple, ya que sólo se dedica a recibir datos procedentes de un ordenador para transmitirlo a los demás.



* + - * Switch
        + A través de un switch aquella información proveniente del ordenador de origen es enviada al ordenador de destino.
        + El switch distribuye los datos a cada máquina de destino, mientras que el hub envía todos los datos a todas las máquinas que responden.
        + Red local.



* + - * Router
        + El Router se encarga de **reenviar tus paquetes de datos entre distintas redes**, por lo general una local o LAN y una externa con un puerto WAN que puedes utilizar, por ejemplo, para conectarse al ADSL o la fibra y de ahí a Internet.
        + Los pasos del ruteo son los siguientes:

1. Recibe el paquete de datos

2. Busca cuál es la dirección de destino

3. Verifica la tabla de enrutamiento que tiene configurada

4. Procede a enviar el paquete a destino por la mejor ruta posible.

* + - * + A día de hoy los Routers pueden cumplir las funciones de los otros dos dispositivos que hemos mencionado, ya que suelen incluir un Switch (o Hub) de entre 4 y 8 puertos Ethernet. De esta manera pueden hacer lo mismo, sólo que **sin limitarse a las redes locales** y abriendo las puertas a que los ordenadores puedan conectarse también a otras redes externas.
        + Los routers también **incorporan otras tecnologías como un Firewall basado en hardware**, que protege tu red de forma inteligente y sin necesitar instalar nada más de posibles ataques DDoS y otros peligros. Además también incluyen otros servicios básicos como NAT, un servidor DHCP y servidor DNS.
        + Clock rate: velocidad de envío de datos (bits/seg)
        + Tabla de enrutamiento

Una tabla de enrutamiento es un conjunto de reglas que sirven para determinar qué camino deben seguir los paquetes de datos.

Los componentes de una tabla de enrutamiento son:

Red de destino

Corresponde a la red de destino donde deberá ir el paquete de datos.

Máscara de subred

Es la que se utiliza para definir la máscara de subred de la red a la que debemos ir.

Siguiente salto

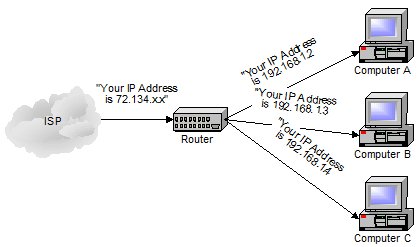
es la dirección de IP de la interfaz de red por donde viajará el paquete de datos, para seguir con su camino hasta el final.

Interfaz de salida

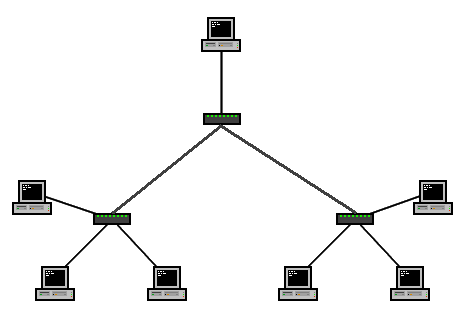
Es la interfaz de red por donde deben salir los paquetes, para posteriormente llegar finalmente al destino.

Métricas

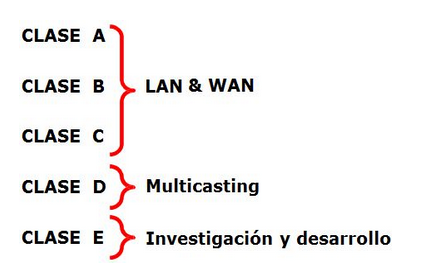
Tienen varias aplicaciones. Una de ellas consiste en indicar el número mínimo de saltos hasta la red de destino, o simplemente el «coste» para llegar hasta la red de destino, y sirve para dar prioridad.



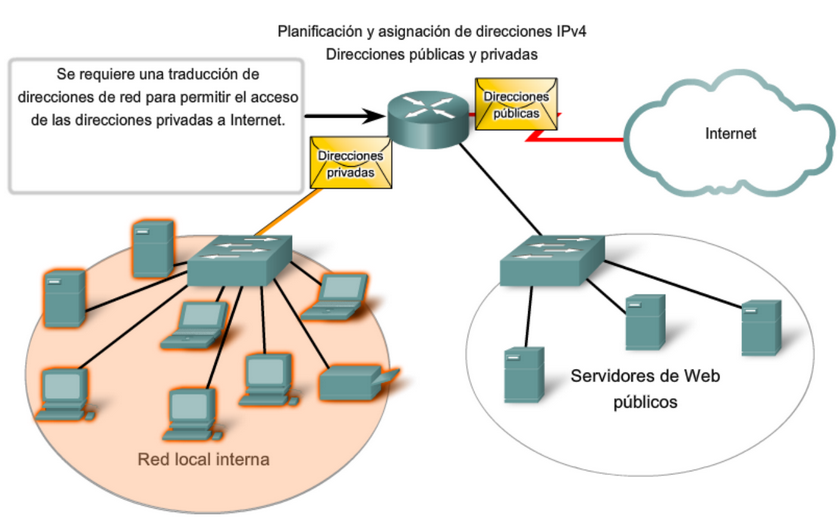
* + En árbol
    - Este tipo de topología es la evolución natural de una red en estrella, permitiendo su extensión o posterior ampliación a distintas zonas del edificio.
    - Las principales características de este sistema son una fusión de las de la topología en bus y en estrella.

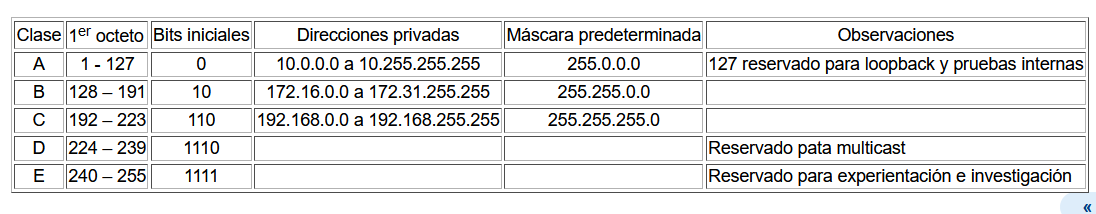


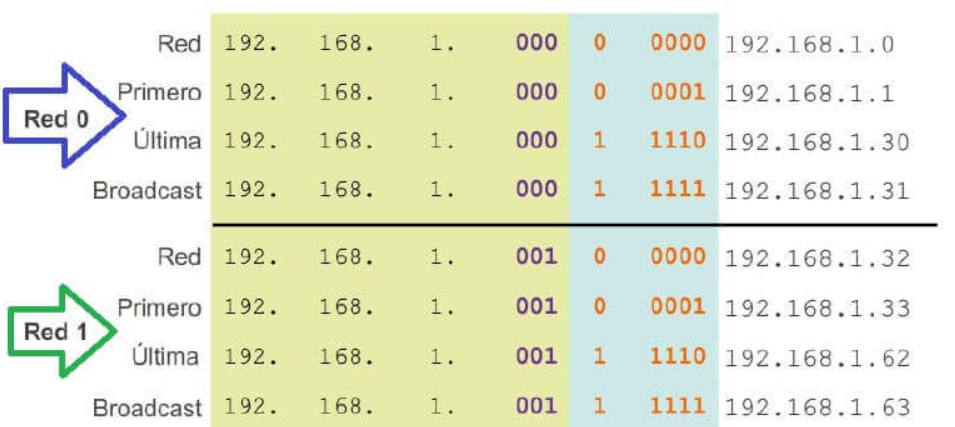
* + <https://ikastaroak.ulhi.net/edu/es/IEA/ICTV/ICTV09/es_IEA_ICTV09_Contenidos/website_32_red_en_anillo.html>
* ¿Qué es un protocolo?
  + Hablamos de un protocolo para referirnos a un sistema de normas que regulan la comunicación entre dos o más sistemas que se transmiten información a través de diversos medios físicos.
  + En otras palabras, no es más que un idioma común que los ordenadores deben entender para comunicarse. Son secuencias de bits debidamente ordenadas para que todos los ordenadores descifren correctamente la información que comparten.
  + Ejemplos de distintos protocolos:
    - FTP (File Transfer Protocol)
      * Protocolos empleados para subir o descargar archivos a altas velocidades a través de una red informática
    - DNS (Domain Name Service)
      * Un protocolo de nombres para las páginas Web de Internet, que permite conectar su dirección URL con la ruta específica en donde se hallan los recursos y la información.
    - IP(Internet Protocol)/TCP (Transmission Control Protocol)
      * El protocolo IP encamina los paquetes de datos a través de la mejor ruta disponible y de una forma segura. No obstante, no garantizan la llegada del paquete a destino.
      * Por ello, para garantizar la llegada del paquete a destino se usa el protocolo TCP.
    - POP, HTTP…
* Dirección IP
  + Todos los ordenadores tienen su propia dirección, conocida como la dirección IP, que no es más que una secuencia de 32 bits que sirve para identificar y encontrar a las computadoras (bien sea un servidor o un ordenador cliente) dentro de una misma red.
  + Hay tres tipos de direcciones dentro del rango de direcciones de cada red IPv4:
    - Dirección de red
      * La dirección de red es una manera estándar de hacer referencia a una red.
      * Todos los hosts dentro de la red comparten la misma dirección de red.
    - Dirección de host
      * Cada dispositivo final requiere una dirección única para comunicarse en la red. En direcciones IPv4, los valores entre la dirección de red y la dirección de broadcast se pueden asignar a los dispositivos finales en una red.
    - Dirección de broadcast
      * La dirección de broadcast IPv4 es una dirección especial para cada red que permite la comunicación a todos los host en esa red. Para enviar datos a todos los hosts en una red a la vez, un host puede enviar un único paquete dirigido a la dirección de broadcast de la red, y cada host en la red que recibe este paquete procesa su contenido.
      * La dirección de broadcast utiliza la dirección más alta en el rango de la red. Ésta es la dirección en la cual los bits de la porción de host son todos 1. Todos 1 en un octeto en forma binaria es igual al número 255 en forma decimal. Por lo tanto, por ejemplo para la red 10.1.1.0/24, en la cual se utiliza el último octeto para la porción de host, la dirección de broadcast sería 10.1.1.255.
  + Existen también diferentes clases dentro de las direcciones IP:

****

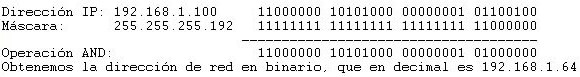
* + Una dirección IP se divide en dos partes: porción de red (N) y porción de host (H).
    - Respecto a los octetos de la dirección IP:
      * En la clase A se identifica como N.H.H.H
      * En la clase B se identifica como N.N.H.H
      * En la clase C se identifica como N.N.N.H
  + Se pueden encontrar dos tipos:
    - IP pública
      * Son direcciones IP que se utilizan para conectar dispositivos a través de Internet.
      * La **pública** es el identificador de nuestra red desde el exterior, es decir, la de nuestro router de casa, que es el que es visible desde fuera.
    - IP privada
      * Las redes privadas permiten conectar equipos dentro de una infraestructura de red.
      * Ejemplo: cada una de las direcciones IP que el router asigna a nuestro ordenador, móvil, tablet o cualquier otro dispositivo que se esté conectado a él.
      * Otro ejemplo: Intranet.



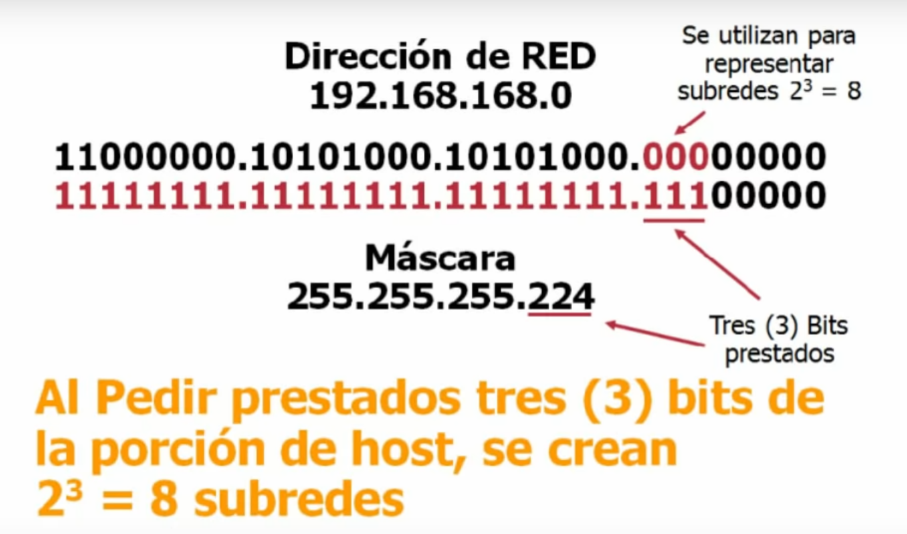
* + Cuando una máquina con una dirección IP privada quiere conectarse a Internet, deberá sustituir esa dirección IP privada en una IP pública. Este proceso es conocido como **NAT** (Network Address Translation)
  + Resumen de clases y rangos en redes publicas y privadas:
* Mascara de red
  + La máscara de red diferencía la porción de red de la porción de hosts en una dirección IP.
  + La parte fija identifica nuestra red, mientras que la parte variable permite identificar a cada uno de los dispositivos conectados en ella. Gracias a la máscara de red podemos distinguir direcciones IP que a simple vista son iguales, pero que al tener diferentes máscaras de red, hace que sean distintas.
  + Ejemplos:
    - 1.
      * Esto quiere decir que si tenemos una máscara de subred 255.255.255.0 y partimos de una dirección IP de nuestro equipo, 192.168.1.X, todas las direcciones de nuestra red tendrán una **parte fija**, en este caso 192.168.1 y un **último octeto que será variable**.
      * Si partimos de una dirección IP 192.168.0.0 y una máscara de subred 255.255.254.0, significa que la red irá **desde la dirección 192.168.0.0 hasta la 192.168.1.255**. Es decir, tendremos disponibles dos subredes con un total de hasta 255 equipos en cada una de ellas.
    - 2.
      * La red 200.33.146.0/30 indica que habrá 2 bits que identifiquen la dirección del host. La máscara de red se podría expresar también de la siguiente manera: 255.255.255.252 (255-2^0-2^1) (255-2^2+1)
      * En total habrá cuatro direcciones distintas y 2 hosts: 200.33.146.0 (red), 200.33.146.1, 200.33.146.2, 200.33.146.3 (broadcast)



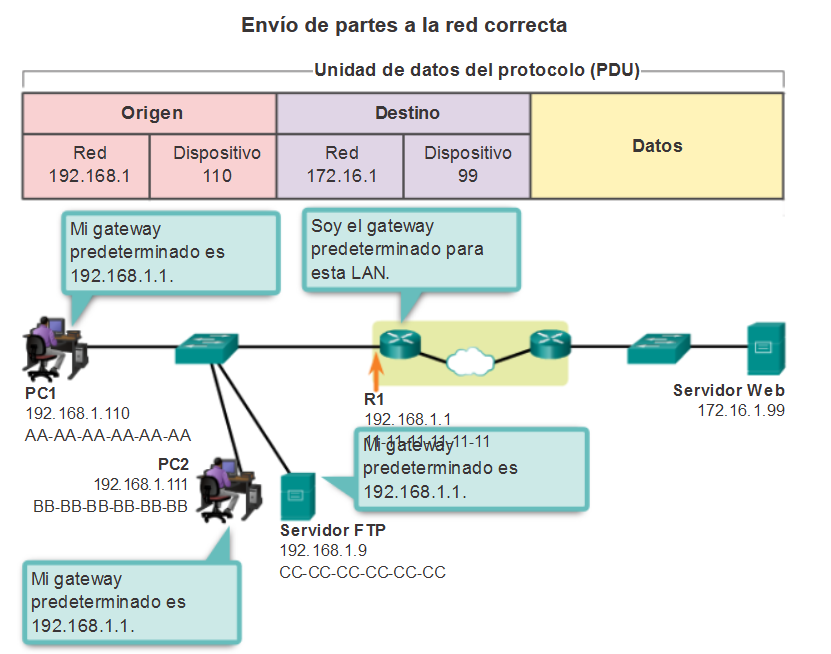
* + Operación AND
    - El objeto de la operación AND es encontrar la dirección de red/subred a la que pertenece un host dado su dirección IP y su máscara de red/subred:



* + Operación OR
    - El objeto de la operación OR es encontrar la dirección de broadcast para la red/subred a la que pertenece un host dado su dirección IP y su máscara de red/subred
  + Subredes (subneting)
    - Objetivo: dividir una red simple en redes múltiples (subredes).
    - Se realiza convirtiendo bits de hosts en bits de red (convirtiendo 0 en 1 en la máscara de red).
    - Ejemplo:
      * Tenemos una dirección de red con una IP 192.168.168.0/24. Queremos dividir esta red en 8 nuevas subredes.



* + - * Las 8 nuevas subredes son las siguientes:
        + 11111111.11111111.11111111.000
        + 11111111.11111111.11111111.001
        + 11111111.11111111.11111111.011
        + 11111111.11111111.11111111.111
        + 11111111.11111111.11111111.110
        + 11111111.11111111.11111111.100
        + 11111111.11111111.11111111.101
        + 11111111.11111111.11111111.010
      * Cada una de las subredes tendrá una **mascara de subred** de 255.255.255.224, con capacidad para albergar 32 direcciones distintas.
* Dirección MAC
  + **La dirección MAC es un identificador único** que cada fabricante le asigna a la tarjeta de red de sus dispositivos conectados.
  + Como son identificadores únicos, las MAC pueden ser utilizadas por un administrador de red para permitir o denegar el acceso de determinados dispositivos a una red.
* Todo el proceso interno desde que entramos en una página web esta en el punto [1.7](#_1.7._Tarea_25).
* Puerta de enlace predeterminada (Default Gateway)
  + Es el nodo que sirve como enlace entre dos redes informáticas, es decir, es aquel dispositivo que conecta y dirige el tráfico de datos entre dos o más redes.
  + Este dispositivo, al conectar dos redes del Internet Protocol (IP), poseerá:
    - Una dirección IP privada: para identificarse dentro de la red local (intranet),
    - Una dirección IP pública: para identificarse dentro de la red exterior (extranet).
  + En casa este dispositivo suele ser el router, que conecta la red local (LAN) con el Internet (WAN).



* Servidor DHCP
  + El protocolo de configuración dinámica de host (DHCP, Dynamic Host Configuration Protocol) es un estándar TCP/IP diseñado para simplificar la administración de la configuración IP de los equipos de nuestra red
  + Si disponemos de un servidor DHCP, la configuración IP de los PCs puede hacerse de forma automática, evitando así la necesidad de tener que realizar manualmente uno por uno la configuración TCP/IP de cada equipo.
  + Utiliza un modelo cliente-servidor en el que el servidor DHCP mantiene una administración centralizada de las direcciones IP utilizadas en la red.
  + El servidor proporciona al cliente los siguientes datos:
    - Dirección IP (obligatoria)
    - Mascara de red (obligatoria)
    - Dirección servidor DNS
    - Dirección puerta de enlace predeterminada
  + SI las IP configuradas son dinámicas, normalmente se “alquilan” por un periodo de tiempo. Cuando se caduca, el cliente vuelve a solicitar una nueva dirección IP.
  + Reserva de direcciones IP
    - Consiste en reservar algunas direcciones IP para asignárselas siempre a los mismos PCs clientes de forma que cada uno siempre reciba la misma dirección IP.
    - Se asigna la dirección IP junto con la dirección MAC.
    - Ejemplos: servidores DNS, servidores web, impresoras…

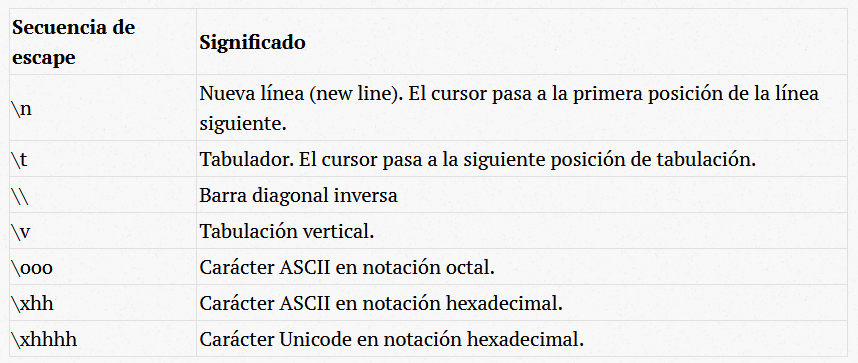
## Tarea 40 – La seguridad es importante

* Malware
  + Se trata de un tipo de software o de aplicación que tiene como objetivo hacer daño al dispositivo en el que se ha conseguido alojar, instalar o infiltrar, ya sea un ordenador, un teléfono móvil o cualquier otro aparato.
  + **El software defectuoso no es malware,** aunque pueda acabar teniendo efectos similares en algunos casos.
  + Tipos de malware
    - * Virus informático
        + Un tipo de malware cuyo objetivo es alterar el correcto funcionamiento de un dispositivo.
        + Necesita ser ejecutado por el usuario pensando que es una aplicación legítima, y una vez lo hace, puede replicarse e infectar el equipo.
      * Gusano informático
        + Malware autosuficiente capaz de ejecutarse y proliferar sin la interacción del usuario. Ni siquiera tiene que estar usando su equipo para que el gusano se active, se replique y se propague.
        + La diferencia fundamental entre un gusano y un virus es la forma en la que aquel propaga copias de sí mismo a máquinas no infectadas. Los virus necesitan usar la programación o el código de su equipo para ejecutarse y replicarse, mientras que los gusanos son autónomos.
      * Troyano
        + Es un malware que va dentro de un programa legítimo o disfrazado de él para introducirse en tu equipo como si usara un Caballo de Troya, de ahí su nombre.
        + Mientras que un virus suele ser destructivo, un troyano trata de pasar desadvertido mientras accede a tu dispositivo con la intención de ejecutar acciones ocultas con las que abrir una puerta trasera para que otros programas maliciosos puedan acceder a él o robar información.
      * Spyware
        + Suele trabajar a escondidas intentando no ser detectado para recolectar información sobre el usuario u organización dueña de un ordenador de forma no autorizada.
        + Se instala en tu equipo por sí sólo o mediante la interacción de una segunda aplicación que lo lanza sin que te des cuenta.
      * Adware
        + Una aplicación en la frontera del malware, porque no siempre es dañino para el ordenador.
        + Su única misión es la de meterse en tu ordenador y empezar a mostrarte publicidad, ya sea mientras estás navegando por internet, a forma de popup en momentos aleatorios o durante la ejecución de un programa.
        + Se instala en tu equipo normalmente a través del proceso de instalación de otras aplicaciones.
      * Ransomware
        + Un malware que secuestra los datos de tu ordenador bloqueándolos sólo ellos o todo el equipo, y pidiéndote un rescate económico a cambio de recuperarlos.
      * DDoS
        + El servidor donde se ejecuta una aplicación solo puede procesar una cierta cantidad de solicitudes de una vez, de modo que si un atacante sobrecarga el servidor con muchas solicitudes, el servidor no dará más de sí. El servidor se caerá y nadie podrá disfrutar del servicio que ofrece.
      * Inyección SQL
        + Son fragmentos de código que generalmente el cracker introduce utilizando una puerta de entrada que siempre está accesible como son los campos de los formularios de cualquier aplicación web (por ejemplo, el buscador de google es un campo de un formulario, formularios para el inicio de sesión, formularios de contacto, ...) para obtener información de las bases de datos o alterarlas.
        + Es necesario poner filtros de seguridad detrás de todo formulario e incluso cifrar la información relevante escrita en las bases de datos por si alguien externo a la organización accediese a ellas.
      * Man in the middle
        + Ocurre cuando una comunicación entre dos sistemas es interceptada por una entidad externa
      * XSS (Cross-site Script)
        + Cuando el atacante es capaz de inyectar un script, normalmente Javascript, en la salida de una aplicación web de forma que esta se ejecutará en el navegador del cliente. Por ejemplo, una ventana emergente que enlace a otra aplicación web.
      * Ingeniería social
        + Trata del arte de manipular a las personas para que compartan su información confidencial: contraseñas, información bancaria, ...
        + Ejemplo: phishing

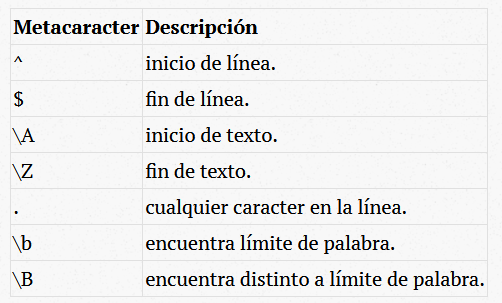
## Tarea 41 – Tu primera expresión regular

**Expresiones regulares/Regex**

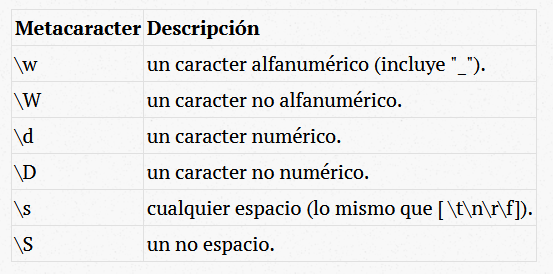
* Son patrones utilizados en programación para encontrar una determinada combinación de caracteres dentro de una cadena de texto.
* Entre sus funciones, destacan:
  + Búsqueda de elementos
  + Reemplazar
  + Validar una contraseña o correo electrónico
    - Para verificar que una contraseña cumpla con ciertos requisitos como una combinación de mayúsculas y minúsculas, dígitos…
  + Automatizar ciertas acciones
  + Reformatear texto
  + Buscar coincidencias entre diferentes textos.
  + Técnicas de scraping
    - Consiste en extraer datos de una o varias páginas web.
    - Por ejemplo, queremos descargar todos los resultados de competiciones deportivas del último fin de semana. Hacerlo de forma manual sería un trabajo titánico. Con web scraping programamos un bot crawler que entra en la web, hace raspado de resultados, y los copia directamente a una base de datos.
    - El uso del scraping, a pesar de ser una técnica desconocida para muchas empresas, es mucho más habitual de lo que se pueda pensar. Algunos autores hablan de que más del 45% del tráfico de la red está realizado por robots y no por humanos.
* Existen varios tipos de sintaxis dentro de estas expresiones (Python)
  + **Literales**
    - Cualquier carácter se encuentra a sí mismo, a menos que se trate de un metacaracter con significado especial.
    - Ejemplo: “Inteligencia”
  + Secuencias de escape
    - La sintaxis de las expresiones regulares nos permite utilizar las secuencias de escape que ya conocemos de otros lenguajes de programación para esos casos especiales como ser finales de línea, tabs, barras diagonales, etc.



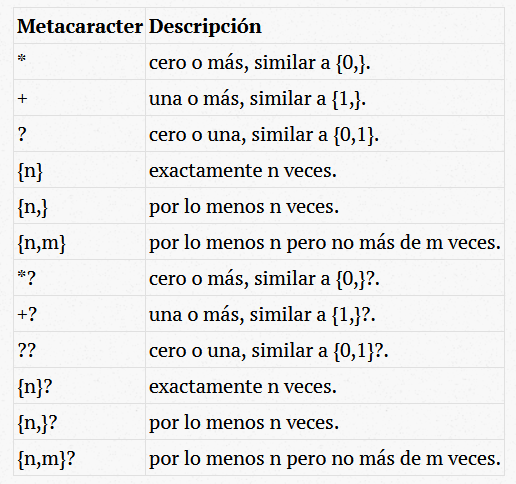
* + Clases de caracteres
    - Se pueden especificar clases de caracteres encerrando una lista de caracteres entre corchetes [], la que encontrará uno cualquiera de los caracteres de la lista.
    - Si el primer símbolo después del "[" es "^", la clase encuentra cualquier carácter que no está en la lista.
  + Metacaracteres
    - Son caracteres especiales.
    - Existen distintos tipos:
      * Delimitadores
        + Permite delimitar dónde queremos buscar los patrones de búsqueda.



* + - * Clases predefinidas
        + Facilitan el código que se debe escribir.



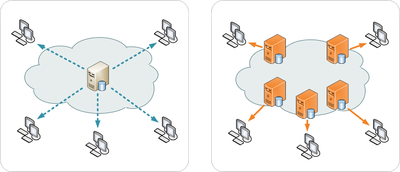
* + - * Iteradores
        + Sirven para especificar el número de ocurrencias del carácter previo, de un metacaracter o de una subexpresión.



* + - * Alternativas
        + Se puede especificar una serie de alternativas para una plantilla usando "|" para separarlas.
        + Por ejemplo, escribiendo do|re|mi encontrará cualquier "do", "re", o "mi" en el texto de entrada.
        + Las alternativas son evaluadas de izquierda a derecha, por lo tanto la primera alternativa que coincide plenamente con la expresión analizada es la que se selecciona.

## Tarea 42 – HTML5, la revolución de los navegadores

* Archivos HTML
  + **HTML** (Lenguaje de Marcas de Hipertexto, del inglés HyperText Markup Language*)* es el componente más básico de la Web.
  + Define el significado y la estructura del contenido web.
  + HTML utiliza "marcas" para etiquetar texto, imágenes y otro contenido para mostrarlo en un navegador Web. Las  marcas HTML incluyen "elementos" especiales como [<head>](https://developer.mozilla.org/es/docs/Web/HTML/Element/head), [<title>](https://developer.mozilla.org/es/docs/Web/HTML/Element/title), [<body>](https://developer.mozilla.org/es/docs/Web/HTML/Element/body), [<header>](https://developer.mozilla.org/es/docs/Web/HTML/Element/header), [<footer>](https://developer.mozilla.org/es/docs/Web/HTML/Element/footer), [<article>](https://developer.mozilla.org/es/docs/Web/HTML/Element/article), [<section>](https://developer.mozilla.org/es/docs/Web/HTML/Element/section), [<p>](https://developer.mozilla.org/es/docs/Web/HTML/Element/p), [<div>](https://developer.mozilla.org/es/docs/Web/HTML/Element/div), [<span>](https://developer.mozilla.org/es/docs/Web/HTML/Element/span), [<img>](https://developer.mozilla.org/es/docs/Web/HTML/Element/img), [<aside>](https://developer.mozilla.org/es/docs/Web/HTML/Element/aside), [<audio>](https://developer.mozilla.org/es/docs/Web/HTML/Element/audio), [<canvas>](https://developer.mozilla.org/es/docs/Web/HTML/Element/canvas), [<datalist>](https://developer.mozilla.org/es/docs/Web/HTML/Element/datalist), [<details>](https://developer.mozilla.org/es/docs/Web/HTML/Element/details), [<embed>](https://developer.mozilla.org/es/docs/Web/HTML/Element/embed), [<nav>](https://developer.mozilla.org/es/docs/Web/HTML/Element/nav), [<output>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/output), [<progress>](https://developer.mozilla.org/es/docs/Web/HTML/Element/progress), [<video>](https://developer.mozilla.org/es/docs/Web/HTML/Element/video), [<ul>](https://developer.mozilla.org/es/docs/Web/HTML/Element/ul), [<ol>](https://developer.mozilla.org/es/docs/Web/HTML/Element/ol), [<li>](https://developer.mozilla.org/es/docs/Web/HTML/Element/li) y muchos otros.
* Archivos CSS
  + **Hojas de Estilo en Cascada** (del inglés **C**ascading **S**tyle **S**heets) o **CSS** es el lenguaje de estilos utilizado para describir la presentación de documentos HTML.
  + Es utilizado para diseñar y dar estilo a las páginas web, por ejemplo, alterando la fuente, color, tamaño y espaciado del contenido, dividirlo en múltiples columnas o agregar animaciones y otras características decorativas
  + CSS describe como debe ser renderizado el elemento estructurado en la pantalla, en papel, en el habla o en otros medios.
  + Se utilizan como selectores los “id” o “class” definidos en las marcas del archivo HTML.
* JavaScript
  + **JavaScript (JS)** es un lenguaje de programación ligero, interpretado, o compilado justo-a-tiempo (just-in-time) con funciones de primera clase.
  + Si bien es más conocido como un lenguaje de scripting (secuencias de comandos) para páginas web, y es usado en muchos entornos fuera del navegador, tal como Node.js, Apache CouchDB y Adobe Acrobat.
  + Permite la nteracción dinámica y usable con el usuario (retweets, botones para compartir contenido, actualización automática de información, ...).
  + Es el lenguaje que se usa para insertar las APIs en la página web.
* CDN
  + Una red de entrega de contenido (CDN) se refiere a un grupo de servidores distribuidos geográficamente que trabajan juntos para ofrecer una entrega rápida de contenido de Internet (es la 2. configuración en la foto).



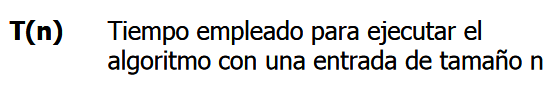
* + Tiene el fin de maximizar el ancho de banda para el acceso a los datos de clientes por la red.
  + Una CDN permite la transferencia rápida de activos necesarios para cargar contenido de Internet, incluyendo páginas HTML, archivos javascript, hojas de estilo, imágenes y vídeos. La popularidad de los servicios de CDN sigue creciendo y hoy la mayoría del tráfico web se lleva a cabo a través de CDN, incluido el tráfico de los principales sitios como Facebook, Netflix y Amazon.

## Tarea 44 – Análisis del rendimiento de las aplicaciones de IA / Tarea 45 – Algoritmos de búsqueda

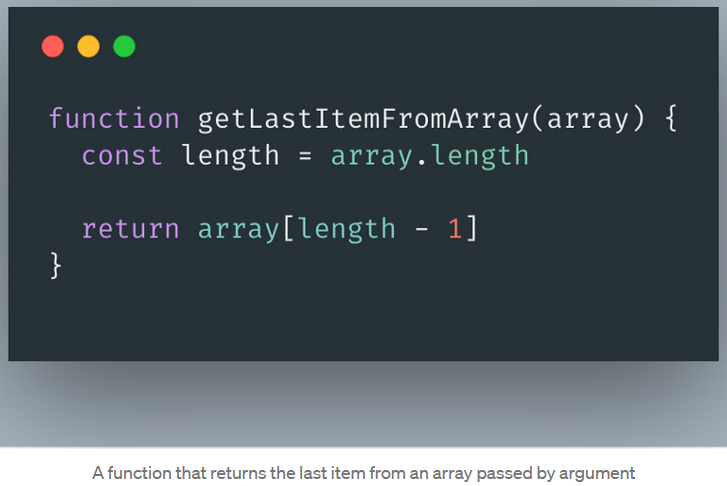
* Latencia
  + Es el tiempo que tarda en transmitirse un paquete dentro de la red.
  + Como medir la latencia:
    - Ping
      * Medido en milisegundos.
      * Mide el tiempo que tardan en comunicarse una conexión local con un equipo remoto en la red IP.
  + Factores que generan latencia
    - La tecnología de acceso a Internet.



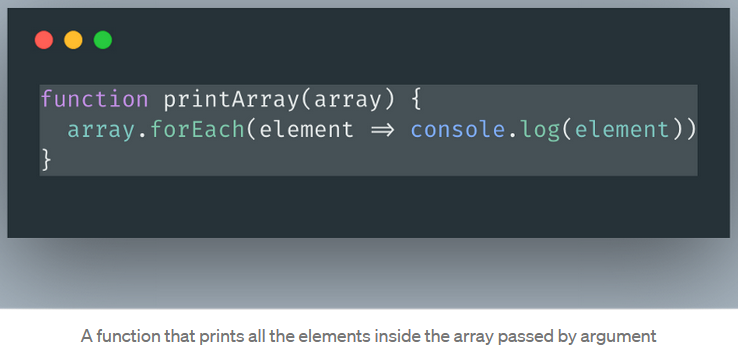
* + - La distancia entre los dos puntos que quieran establecer la comunicación.
    - Capacidad del dispositivo desde el que nos conectamos.
* Eficiencia algorítmica
  + Medida del uso de los recursos computacionales requeridos por la ejecución de un algoritmo en función del tamaño de las entradas.



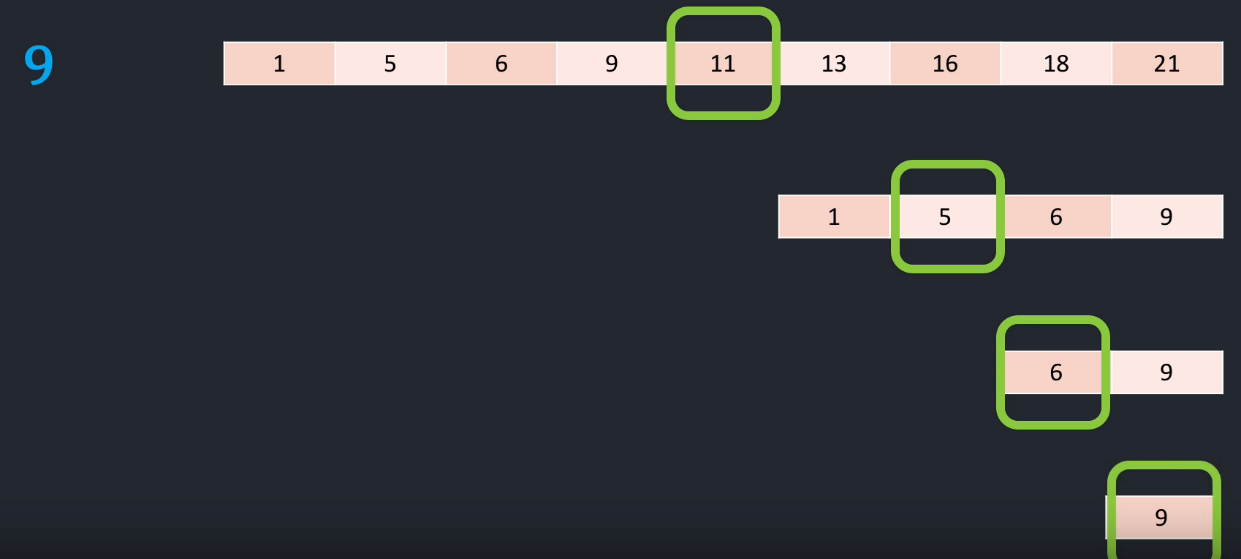
* Notación Big-O
  + ¿Qué es?
    - Técnica/Metodología para planificar el comportamiento de los algoritmos y medir su rendimiento y eficiencia (en función del volumen de datos).
  + ¿Cómo funciona?
    - Mide el numero de operaciones para resolver el problema dado.
    - Cada uno de los pasos es una unidad básica de cálculo.
    - Independiente del programa/computadora.
  + Términos de complejidad más utilizados
    - O(1): Complejidad constante
      * Independiente del tamaño de entrada o salida
      * El tiempo de ejecución y recursos utilizados por el algoritmo siempre será el mismo.



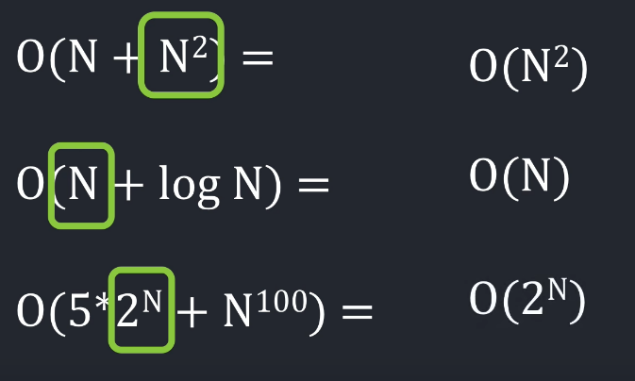
* + - O(n)
      * Relación lineal entre el tamaño del valor de entrada y el tiempo de ejecución.
      * Similitud con leer un libro
        + La duración de la lectura será proporcional al número de hojas.



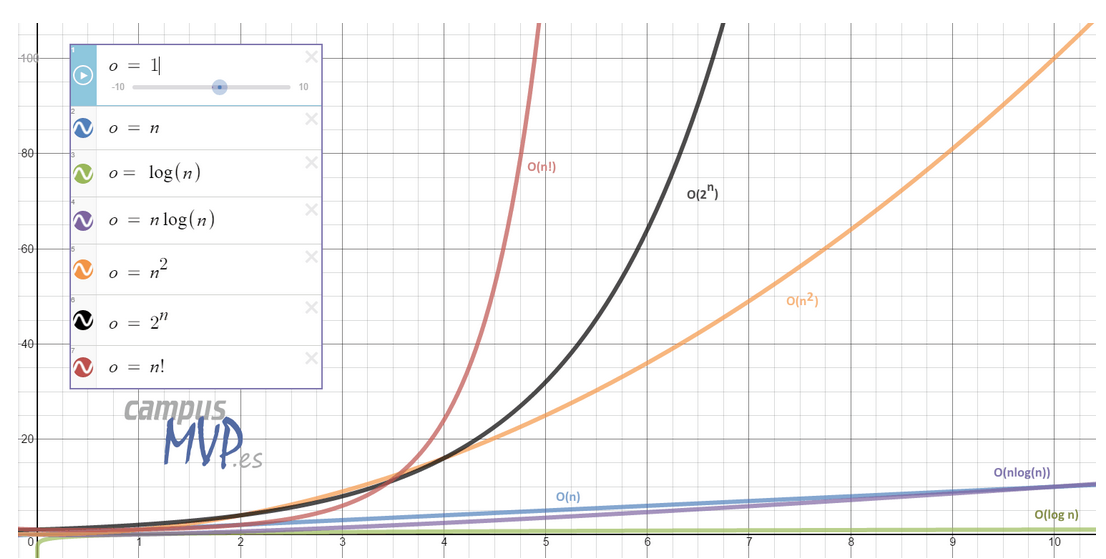
* + - O(log n)
      * Cuando el tiempo de ejecución o uso de recursos es directamente proporcional al resultado logarítmico del tamaño del valor de entrada.
        + Si tenemos un dato de entrada cuyo tamaño es 10, nos toma 1 segundo en la implementación del algoritmo.
        + Si son 200, serán necesarios 2 segundos de ejecución.
      * Por regla general se asocia con algoritmos que "trocean" el problema para abordarlo.
      * Ejemplo: Búsqueda binaria



* + - O(n^2)
      * El rendimiento es directamente proporcional al cuadrado del tamaño del valor de entrada.
      * Cuando hay que hacer una iteración por todos los elementos de la lista con cada uno de los elementos
      * Ejemplo: algoritmo de ordenación de burbuja
    - O(2^n)
      * Cuando su rendimiento se incrementa al doble cada vez que se agregue un nuevo dato al valor de entrada
      * Ejemplo:
        + Comparación. Tenemos una lista de 10 elementos (2^10) que pueden tener valor 1 o 0. Para comprobar cual de todas posibilidades ofrece el mayor resultado, se comprueban todas las posibilidades posibles.
        + Sucesión de Fibonacci.
  + Términos de complejidad no dominantes

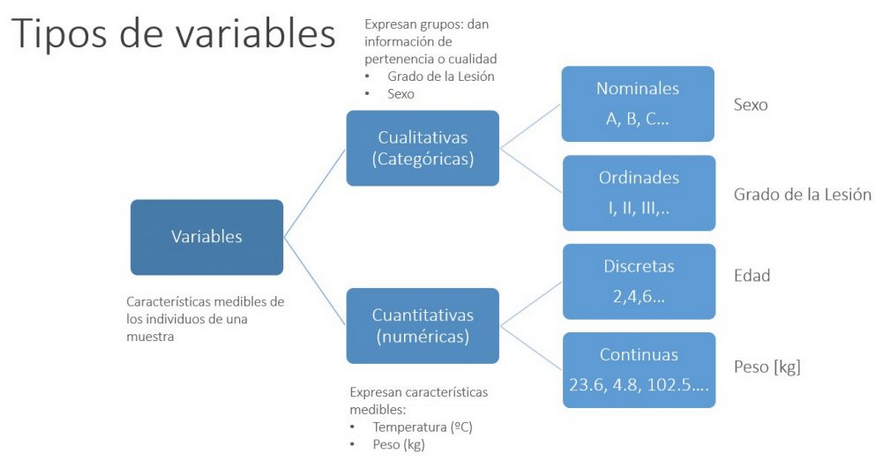


* + Gráfico del rendimiento del algoritmo vs el número de datos

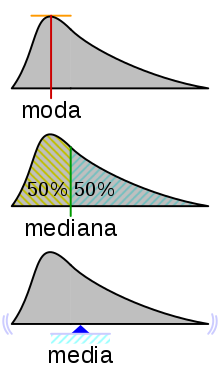


## Tarea 46 – Estadística descriptiva para novatos

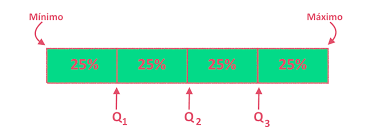
* Etapas del proceso de análisis de datos
  + Problema
    - El análisis de datos nace de una necesidad real, un problema.
    - En esta etapa se debe definir el foco y el objetivo del estudio.
  + Recolección
    - Es el diseño de un método para obtener datos.
    - Puede ser interesante definir el tipo de variable para cada dato que se quiera recolectar.



* + - Ejemplo: encuestas, pruebas en laboratorios…
  + Limpieza
    - Homogeneizar los datos en cuanto a formato.
    - Eliminar datos incorrectos, inútiles, atípicos…
  + Exploración
    - Explorar el dato mediante la estadística descriptiva (medias, histogramas, varianzas…)
  + Análisis
    - Realizar contrastes de hipótesis para definir si por ejemplo existe una correlación, si un grupo es distinto a otro, si existen patrones…
  + Conclusión
    - Interpretar los resultados del análisis y la exploración y sacar las conclusiones principales.
* Métricas estadísticas clave
  + Media
    - Valor promedio de un conjunto de datos numéricos.
    - Es la suma del conjunto de valores dividida entre el número total de valores.
  + Mediana
    - La mediana de un conjunto de números es el número medio en el conjunto (después que los números han sido ordenador de menor a mayor).
    - Es decir, la mitad de los números son superiores a la mediana y la mitad de los números tienen valores menores que la mediana.
  + Moda
    - Es el valor más repetido (más frecuente) del conjunto de datos.



* + Cuartiles
    - Los cuartiles son medidas estadísticas de posición que tienen la propiedad de dividir la serie estadística en cuatro grupos de números iguales de términos.
    - Los cuartiles son:
      * Q1 (25% de los datos)
      * Q2 (50% de los datos) == Mediana
      * Q3 (75% de los datos)



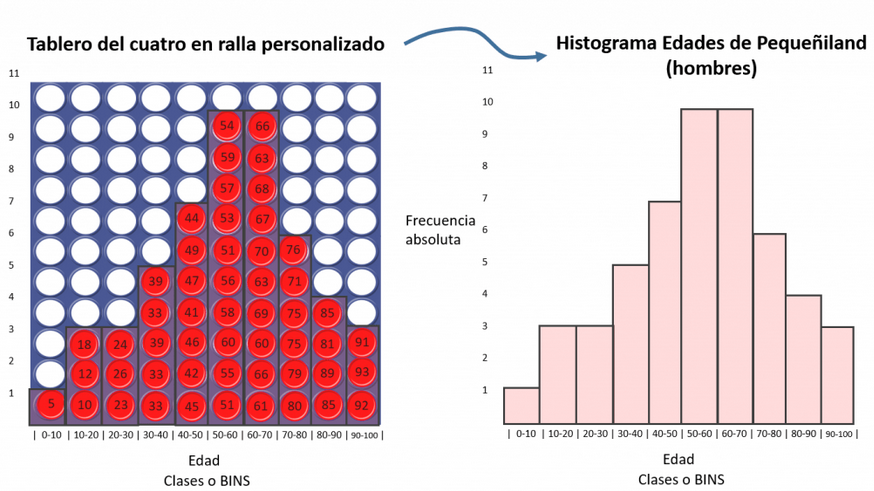
* + Varianza
    - La Varianza es una medida de dispersión que se utiliza para representar la variabilidad de un conjunto de datos respecto de la media aritmética de los mismo.
    - Se calcula como la suma de los residuos elevados al cuadrado y divididos entre el total de observaciones.



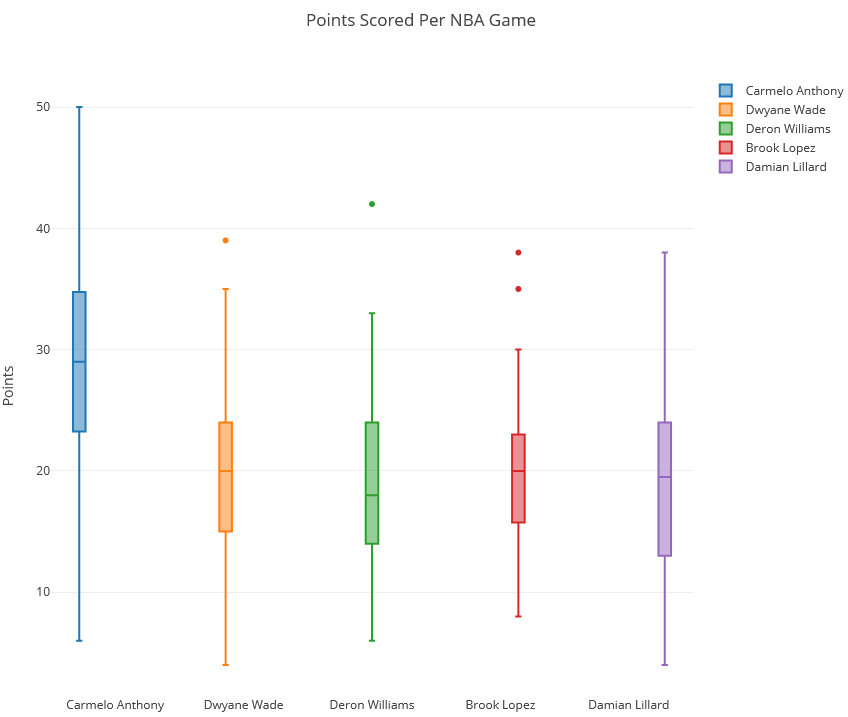
* + Desviación típica
    - La desviación típica es la desviación media de una variable respecto de su media aritmética.
    - Sirve para cuantificar la variación o la dispersión de datos numéricos en una variable aleatoria, población estadística, conjunto de datos o distribución de una probabilidad.
    - Cálculo:
      * Raíz cuadrada de la varianza.



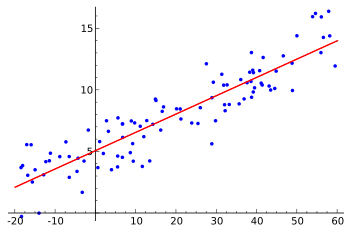
* Gráficas
  + Histograma
    - Diagrama de barras para medir la frecuencia de rangos.



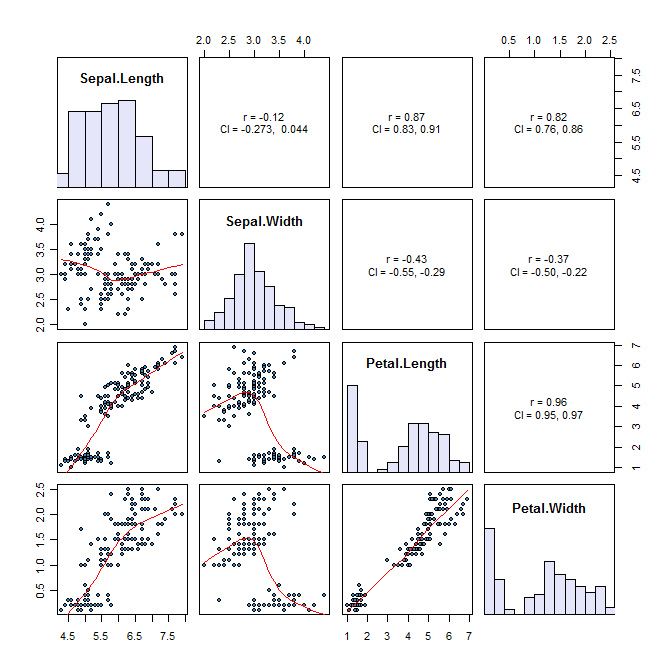
* + Boxplot
    - Permite visualizar las medidas de dispersión de cada grupo de valores.
    - Se basa en los cuartiles.



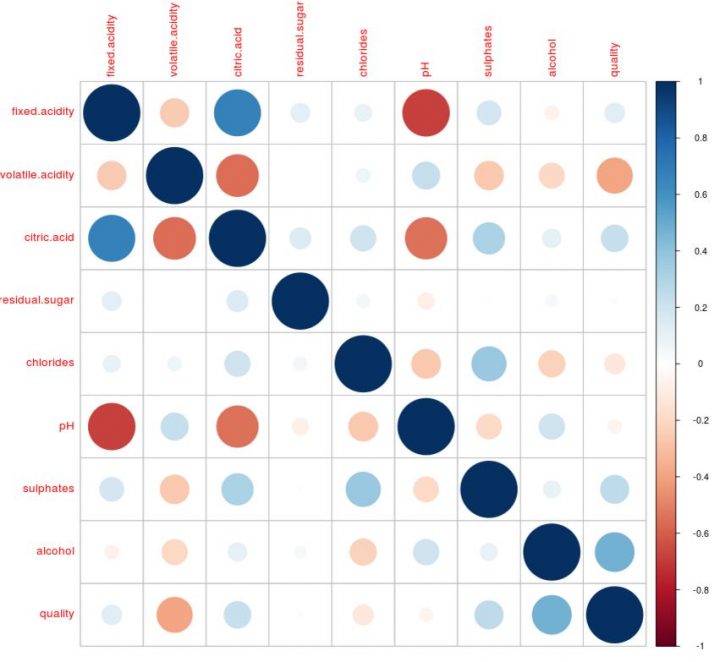
* + Scatter
    - Es una nube de puntos sin unir mediante una línea.
    - Es útil para intuir si existe correlación entre las dos variables.



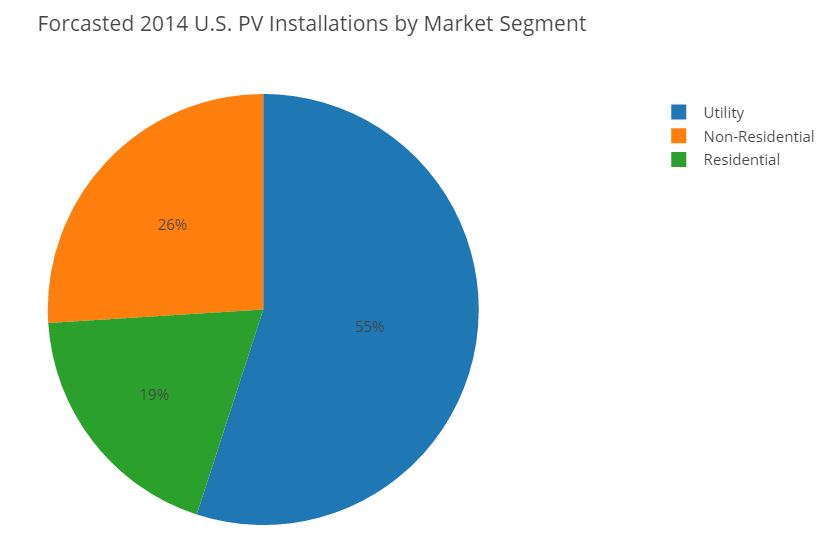
* + Matrix plot
    - Se utiliza para graficar 3 o más variables entre sí.
    - Relaciona una variable con las otras.



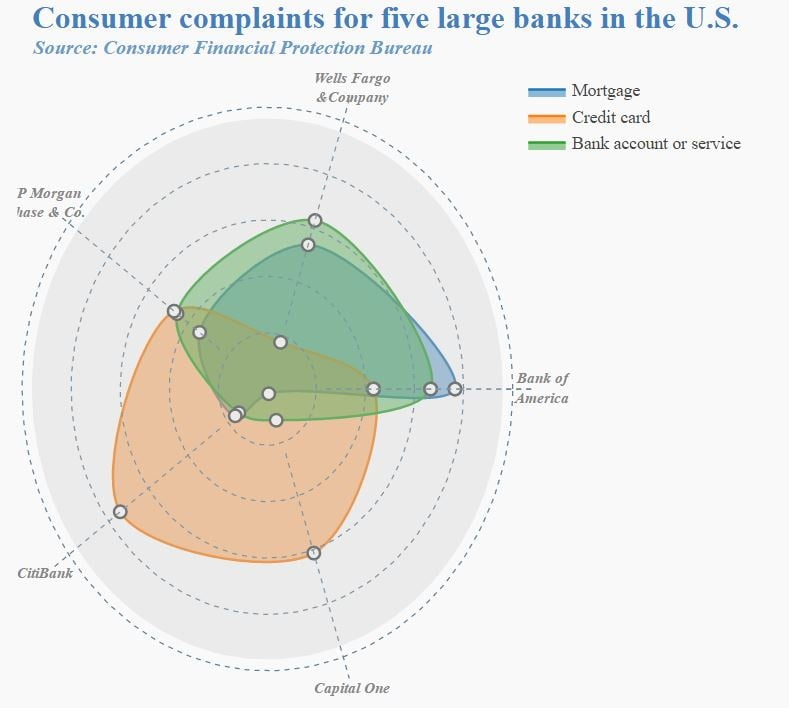
* + Mapa de correlaciones
    - Indica en mediante la intensidad del color las variables que están más correlacionadas que las otras.



* + Diagrama de sectores
    - Para expresar porcentajes y proporciones.

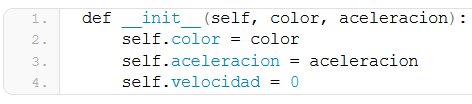


* + Gráfico radial
    - Los gráficos radiales son una forma de comparar múltiples variables cuantitativas.
    - La visualización se hace en forma de polígono.

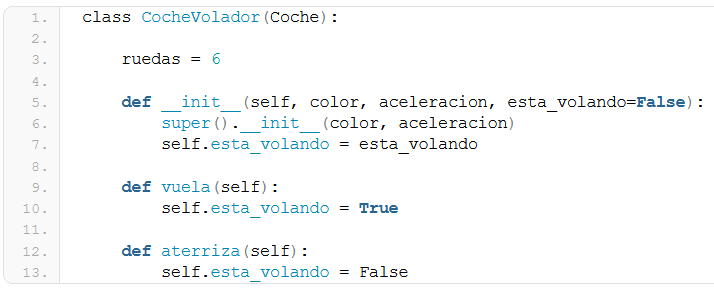


## Tarea 49 – Aprender a pensar como un programador: Introducción a la POO

* Ámbitos en Python
  + Ámbito local
    - Corresponde con el ámbito de una función, que existe desde que se invoca a una función hasta que termina su ejecución.
    - No se puede acceder a las variables de una función desde fuera de esa función o desde otra función.
  + Ámbito global
    - Corresponde con el ámbito que existe desde el comienzo de la ejecución de un programa.
    - Todas las variables definidas fuera de cualquier función corresponden al ámbito global, que es accesible desde cualquier punto del programa, incluidas las funciones.
    - Si desde un módulo A importamos un módulo B mediante un “import”, desde A podremos acceder a las variables globales de B.
* Programación orientada a objetos (POO)
  + Es un paradigma de programación que usa objetos y sus interacciones para diseñar aplicaciones y programas de computadora.
  + Se basa en el modelo objeto, donde el elemento principal es el objeto, el cual es una unidad que contiene todas sus características y comportamientos en sí misma, lo cual lo hace como un todo independiente, pero que se interrelaciona con objetos de su misma clase o de otras clase, como sucede en el mundo real.
  + Conceptos importantes:
    - Clase
      * Definición de las propiedades y comportamiento de un tipo de objeto concreto.
    - Objeto
      * Es una concreción o instancia de una clase.
      * Ejemplo: el VW Polo que tenemos es el objeto y pertenece a la clase “coche”.
    - Constructor
      * Método especial “\_\_init\_\_()”
      * Para crear atributos iniciales de un objeto.
    - Atributo
      * Definen el estado del objeto.
      * Ejemplo: altura, peso, edad…



* + - Método
      * Las funciones definidas dentro de las clases.
  + Herencia en Python
    - Es la capacidad de reutilizar una clase extendiendo su funcionalidad.
    - Una clase que hereda de otra puede añadir nuevos atributos, ocultarlos, añadir nuevos métodos o redefinirlos.
    - La herencia puede ser múltiple (más de una clase)
    - Ejemplo:



* + - * La clase “CocheVolador” hereda de la clase “Coche”
      * La clase “CocheVolador” redefine el atributo de clase “ruedas”, estableciendo su valor a 6 e implementa dos métodos nuevos: “vuela()” y “aterriza()”.
  + Encapsulación: atributos privados
    - Es la capacidad que tiene un objeto de ocultar su estado, de manera que sus datos solo se puedan modificar por medio de las operaciones (métodos) que ofrece.
    - Por defecto, todos los atributos de una clase son públicos (Python).
    - Para hacerlo privado:
      * 1. metodo
        + Self.\_\_x (doble guion bajo)
      * 2. metodo: “getters” y “setters”
        + Son decoradores y se identifican por tener un
  + Polimorfismo
    - es la capacidad de una entidad de referenciar en tiempo de ejecución a instancias de diferentes clases.
    - Ejemplo:



* + - Se ha definido una función llamada “a\_cantar()”. La variable animal que se crea dentro del bucle “for” de la función es polimórfica, ya que en tiempo de ejecución hará referencia a objetos de las clases “Perro”, “Gato” y “Vaca”. Cuando se invoque al método “sonido()”, se llamará al método correspondiente de la clase a la que pertenezca cada uno de los animales.

## Tarea 52 – Estructuras de datos

* Estructuras de datos
  + Es una forma particular de organizar datos en una computadora para que puedan ser utilizados de manera eficiente.
  + Las estructuras de datos eficientes son clave para diseñar algoritmos eficientes.
* Algunos tipos de estructuras de datos
  + Array
    - Almacena los datos “un elemento al lado del otro”.
    - Pueden ser unidimensionales (x), bidimensionales (x\*y) o tridimensionales (x\*y\*z).

Forma

Descripción generada automáticamente con confianza media

* + Listas enlazadas
    - Cada elemento apunta al siguiente excepto el último que no tiene sucesor y el valor del enlace es “null”.
    - Los elementos son registros que contienen el dato a almacenar y un enlace (dirección de memoria) al siguiente elemento.

Diagrama

Descripción generada automáticamente

* + Pilas
    - Son listas que tienen la particularidad de sólo poder eliminar o insertar en la cima de la lista.
    - LIFO (Last In First Out).

Interfaz de usuario gráfica

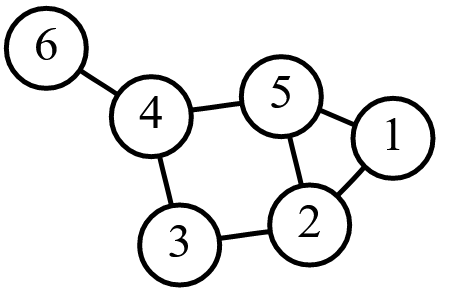
Descripción generada automáticamente con confianza baja

* + Colas
    - Es una secuencia de elementos donde la operación de inserción se realiza por un extremo y la operación de extracción por el otro.
    - FIFO (First In First Out).

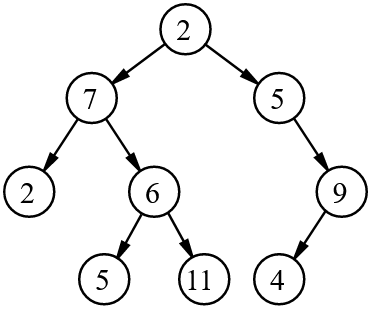
Pantalla de video juego

Descripción generada automáticamente con confianza baja

* + Grafos
    - Es una estructura de datos no lineal ni jerárquica.
    - Ejemplo: red de conexiones de vuelos de Ryanair.



* + Árboles
    - Es una estructura de datos que imita la estructura jerárquica de un árbol, con un valor en la raíz y subárboles con un nodo padre.



* Operaciones con estructuras de datos
  + Búsqueda
    - Implica la localización de un registro caracterizado por una determinada clave o también el acceso a todos los registros que cumplan una o más condiciones.
  + Inserción
    - Se trata de añadir nuevos registros a la estructura de datos.
  + Eliminación
    - Es la operación de borrado de un registro es la estructura de datos.
  + Ordenación
    - Es la operación de clasificar los registros conforme a un orden lógico determinado (por ejemplo, alfabéticamente).

## Tarea 78 – Aprender a programar en un abrir y cerrar de ojos

* Python
  + ¿Qué es?
    - Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional.
  + Comentarios
    - Los comentarios de una sola línea comienzan con el carácter almohadilla (#).
    - Los de múltiples líneas comienzan y finalizan con triple comilla doble (""").
  + Indentación
    - Las llaves de indentación utilizadas en otros lenguajes de programación son sustituidas por un sangrado de 4 espacios en blanco.

Gráfico, Gráfico de embudo

Descripción generada automáticamente

* + Operadores matemáticos

Tabla

Descripción generada automáticamente

* + Operadores lógicos

Tabla

Descripción generada automáticamente

* + Tipos de datos
    - Los datos se componen de:
      * Identificador
        + Nombre para referenciar el dato dentro del software.
      * Tipo
        + Rango de valores que puede tomar el dato y su ocupación en memoria durante la ejecución del programa.
      * Valor
        + Elemento determinado dentro del rango de valores permitido por el tipo de dato permitido.
    - Existen los siguientes tipos:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

## Tarea 92 – Empezar a trabajar con librerías en Python

* ¿Qué es una librería?
  + Una librería de programación es un conjunto de módulos/funciones que alguien escribió en alguna parte del mundo y lo ha puesto disponible para que cualquiera pueda utilizarlo de forma gratuita.
  + Cada librería tiene un conjunto de módulos con una finalidad específica y que debe ser invocada por el usuario al querer usarlo.
* Tipos de librerías para IA
  + Visualización, cálculo numérico, Machine Learning (para resolver problemas de clasificación y regresión), Deep Learning (orientadas a redes neuronales), procesamiento de Lenguaje Natural…

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

* ¿Cómo utilizar una librería en Python?
  + import <modulo>
    - Para importar el módulo completo.
  + import <modulo> as <pseudónimo>
    - Para importar el módulo completo y llamarlo con otro nombre.
  + from <modulo> import <funciones>
    - Para importar unas determinadas funciones del módulo. A veces es conveniente no importar todo el módulo para no sobrecargar el programa.
  + from <modulo> import \*
    - Para importar todas las funciones del módulo

# BIBLIOGRAFIA

* <https://soundgirls.org/entendiendo-los-convertidores-ad-da/>
* <https://es.slideshare.net/ester.gonzalez/arquitectura-cliente-servidor-104004600>
* <https://platzi.com/blog/que-es-frontend-y-backend/>
* <https://www.diferenciador.com/sistema-digital-y-sistema-analogico/>
* <https://bookdown.org/aquintela/EBE/variables-discretas-y-continuas.html>
* <https://www.youtube.com/watch?v=w1K1InJYZh4>
* <https://www.youtube.com/watch?v=-4rUKlNeCEs>
* <https://platzi.com/blog/que-es-un-etl/>
* <https://www.youtube.com/watch?v=u3Le4IFePnQ&t=8s>
* <https://www.abc.es/tecnologia/consultorio/20150216/abci--201502132105.html?ref=https:%2F%2Fwww.google.com%2F#ancla_comentarios>
* <https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces>
* <https://www.ticportal.es/glosario-tic/base-datos-database>
* <https://www.grapheverywhere.com/bases-de-datos-nosql-marcas-tipos-ventajas/>
* <https://www.oracle.com/es/database/what-is-a-cloud-database/>
* <https://concepto.de/protocolo-informatico/>
* <https://openwebinars.net/blog/que-es-tcpip/>
* <https://www.submarinecablemap.com/>
* <https://gpcinc.mx/blog/redes-lan-man-wan/>
* <https://www.xataka.com/basics/cuales-son-las-diferencias-entre-hub-switch-y-router>
* <https://computerhoy.com/noticias/internet/cuales-son-diferencias-hub-switch-router-43325>
* <https://www.icm.es/2020/02/27/direccion-ip/>
* <https://www.adslzone.net/reportajes/internet/que-es-mascara-de-subred/>
* <http://www.portaleso.com/Redes/Ud_4_redes_V1_c.pdf>
* <https://www.itesa.edu.mx/netacad/introduccion/course/module8/8.1.2.3/8.1.2.3.html>
* <https://docs.microsoft.com/es-es/troubleshoot/windows-client/networking/tcpip-addressing-and-subnetting>
* <https://www.quia.com/files/quia/users/istomar/DIPS/mscara_de_subred.html>
* <http://itroque.edu.mx/cisco/cisco1/course/module3/3.3.3.1/3.3.3.1.html>
* <http://formacion.intef.es/pluginfile.php/37388/mod_resource/content/1/PDF_conlogonuevo/2-Servidor-DHCP-y-DNS.pdf>
* <https://www.redeszone.net/2016/11/29/vlans-que-son-tipos-y-para-que-sirven/>
* <https://www.redeszone.net/tutoriales/redes-cable/configurar-enlace-troncal-switch/>
* <https://www.cisco.com/c/en/us/support/docs/lan-switching/inter-vlan-routing/14976-50.html>
* <https://www.tecnologia-informatica.es/calcular-subredes/>
* <https://www.w3schools.com/sql/default.asp>
* <https://relopezbriega.github.io/blog/2015/07/19/expresiones-regulares-con-python/>
* <https://platzi.com/blog/expresiones-regulares-python/>
* <https://docs.python.org/3/library/re.html#module-contents>
* <https://geekflare.com/es/regular-expression-tester/>
* <https://www.xataka.com/basics/malware-que-que-tipos-hay-como-evitarlos>
* <https://www.avast.com/es-es/c-computer-worm>
* <https://developer.mozilla.org/es/docs/Web/HTML>
* <https://developer.mozilla.org/es/docs/Web/CSS>
* <https://developer.mozilla.org/es/docs/Web/JavaScript>
* <https://www.campusmvp.es/recursos/post/Rendimiento-de-algoritmos-y-notacion-Big-O.aspx>
* <https://medium.com/@charlie_fuentes/notacion-big-0-para-principiantes-f9cbb4b6bec8>
* <https://elvex.ugr.es/decsai/algorithms/slides/2%20Eficiencia.pdf>
* <https://conceptosclaros.com/10-graficos-estadisticos-estadistica-descriptiva-3/>
* <https://conceptosclaros.com/como-analizar-datos/>
* <https://www.ine.es/explica/docs/pasos_tipos_graficos.pdf>
* <https://j2logo.com/python/tutorial/programacion-orientada-a-objetos/>
* <https://unipython.com/programacion-orientada-objetos-python/>