

1.1. Introducción a los sistemas informáticos



Índice

Objetivos	4
Diferencia entre analógico y digital	5
Información analógica puede ser.....	5
Información digital puede ser.....	6
Estructura de un sistema informático.....	7
Un ejemplo de transformación analógico-digital.....	7
¿Qué es información?	7
Representación de la información digital	8
¿Qué es un “bit”?	8
Bit, Byte y palabra.....	8
¿Cuándo usar las unidades en “bits” y cuándo en “bytes”?	10
Un ejemplo sencillo.....	11
Diferentes sistemas de numeración	12
Sistema decimal.....	12
Sistema binario	12
Sistema octal	13
Sistema hexadecimal.....	13
Conversión entre sistemas de numeración.....	14
Conversión de cualquier sistema a decimal.....	14
Conversión de decimal a binario	15
Conversión de binario a hexadecimal	15
Conversión de hexadecimal a binario.....	16
Conversión de binario a octal.....	16
Conversión de octal a binario.....	16
Ejercicio a resolver: conversión de números	17
Codificación de la información.....	19
El código ASCII (American Standard Code for Information Interchange)	19
El código EBCDIC (Extended Binary Coded Decimal Interchange Code).....	20
Código BCD (Binary-Coded Decimal).....	21
Ejercicio a resolver: codificación.....	22

¿Qué son los protocolos de datos?	23
¿Qué incluye un protocolo de datos?	23
¿Qué es una "trama de información"?	24
¿Qué es una red de datos?	25
Diferencia entre hardware, firmware y software.....	26
Ordenadores. Funcionamiento básico	28
Un vistazo al entorno Windows10.....	28
Despedida	33
Resumen.....	33

En esta unidad haremos una introducción a los sistemas informáticos en general, repasando algunos conceptos fundamentales y necesarios para comprender las lecciones siguientes. Quizás parte del contenido ya lo sepas y los puntos tratados te sirvan de recordatorio de tus conocimientos, o bien puedes aprovechar para llenar posibles lagunas o dudas que tengas sobre lo explicado en la lección.

También verás que algunos de los conceptos explicados en esta lección son tratados con mayor profundidad más adelante, proporcionando a veces definiciones alternativas, complementando o ampliando las que te damos ahora. Todo ello es para intentar asegurar que las ideas se fijan bien y avanzas paulatinamente en tu aprendizaje.

Objetivos

1. Disponer de los conceptos básicos sobre tecnologías de la información y las comunicaciones que nos permitan aprovechar y estudiar cómodamente el resto del módulo.
2. Resolver posibles dudas de conceptos básicos que puedan existir.
3. Practicar con algunos ejercicios básicos que nos permitan familiarizarnos con la forma en la que manejamos la información.

Diferencia entre analógico y digital

Seguro que estás acostumbrado a manejar estos dos términos pero, ¿sabes realmente lo que significan? Vamos a verlo.

Información analógica

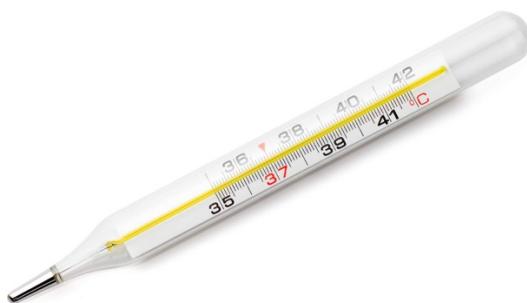
Decimos que una variable (una medida, una señal, etc.) es “**analógica**” cuando puede tomar cualquier valor dentro de su rango de variación, es decir, cuando va cambiando de un valor a otro a lo largo del tiempo lo hace de forma “continua” y para pasar de un valor a otro, pasa por todos los valores intermedios entre ambos.

Información digital

Por otro lado, una variable (medida, señal, etc.) diremos que es “**digital**” cuando en su variación a lo largo del tiempo solamente toma una serie de valores determinados, pero no todos, sino solamente algunos dentro de su rango de variación.

Información analógica puede ser...

- La medida de temperatura en un termómetro de mercurio.
- La hora en un reloj de sol.
- El sonido de las olas del mar.
- La intensidad (variación) de la luz en un amanecer o un atardecer.
- La fuerza que podemos hacer al apretar algo con la mano.
- Una fotografía hecha con negativo (como las de antes).
- etc.



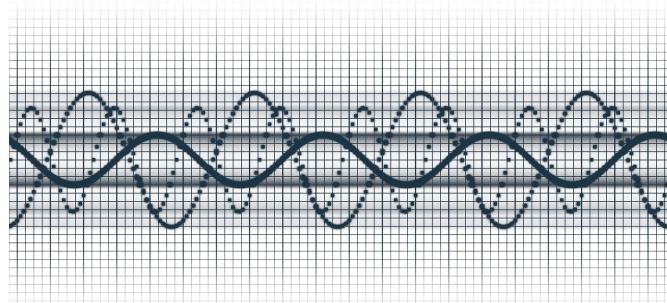
Nota: fíjate que en el caso del termómetro, aunque la columna de mercurio puede recorrer todos los valores de temperatura, nosotros marcamos una escala solamente con una serie de divisiones para medirla. La precisión vendrá dada por la cantidad de divisiones de esa escala de medir.

Información digital puede ser...

- La medida de temperatura en un termómetro digital.
- La hora en un reloj digital moderno.
- Una información escrita alfanumérica.
- La foto de una cámara digital.
- Los pisos de un edificio en los que para un ascensor.
- etc.



Como habrás visto con el ejemplo del termómetro y los demás, en la mayoría de los casos **las magnitudes en la naturaleza son analógicas**, pero nosotros las “medimos”, y al hacerlo normalmente elegimos quedarnos solamente con una serie de valores de una escala de medida. Dependiendo de la resolución de la “regla” con la que medimos tendremos más precisión o menos en la medida.



Es decir, a menudo lo que hacemos es “**digitalizar**” las magnitudes para poder trabajar más cómodamente con ellas, medirlas, representarlas matemáticamente con números o letras, etc.

Esto es importante que lo tengas en mente, pues nos vamos a adentrar en el funcionamiento de los **ordenadores**, es decir, equipos que **trabajan con información digital** para procesarla y almacenarla, y que cuando es necesario se sirven de otros aparatos y circuitos para transformar la información de analógica a digital y viceversa.

Estructura de un sistema informático

Un ejemplo de transformación analógico-digital

A lo mejor recuerdas cuando las fotografías había que hacerlas con una cámara que tenía un rollo de película que había que revelar para luego hacer las fotos y actualmente, en cambio, podemos hacer las fotos directamente con el teléfono móvil y enviarlas como un fichero.



Pues bien, cuando sacas una foto con el móvil lo que estás es digitalizando una imagen y generando un conjunto de datos (fichero o archivo, lo veremos) con un determinado formato, que luego manejamos e intercambiamos con otros equipos.



¿Qué es información?

Llegados a este punto vamos a definir lo que es “**información**”, pues este término lo usaremos mucho.

Para nosotros información será cualquier tipo de “dato” o valor que vayamos a manejar. Ahora sabes que puedes tener información analógica e información digital, pero además podemos decir que la información puede ser un conjunto de caracteres escritos (por ejemplo un texto), o numéricos (una ecuación matemática, valores en número, etc.), una imagen, música, también hablamos de información “audiovisual” para referirnos a las imágenes en movimiento con sonido, es decir, que “todo es información” y lo que vamos a aprender es cómo manejan la información los sistemas informáticos (ordenadores) y lo que hacen con ella, que básicamente será:

1. **Procesarla:** es decir transformarla, hacer operaciones matemáticas con ella, modificarla, etc.
2. **Almacenarla:** guardarla en algún sitio y con un determinado formato y orden.
3. **Intercambiarla** con otros sistemas, esto es, transmitir y recibir información, y todo ello a través de lo que llamamos “**redes de comunicaciones**”.

Representación de la información digital

Ya sabes que la **información digital** es aquella que solamente puede tomar una serie de valores determinados, que a su vez podemos representar matemáticamente por ejemplo con números (recuerda el ejemplo del termómetro), o letras, o cualquier otro símbolo que elijamos.

En este punto vamos a hablarte de un tipo concreto de información: la “**información binaria**” o lo que es lo mismo, el caso en que utilizamos los “**bits**” para representar la información.

¿Qué es un “bit”?

Un “bit” es la **unidad más pequeña de información** que vamos a utilizar, esto es, una información que puede tomar solamente **DOS POSIBLES VALORES**, y que normalmente vamos a representar con los símbolos “**0**” (cero) y “**1**” (uno).

Cualquier cosa que solamente pueda tener dos posibles estados podremos representarla con 1 bit, indicando un “0” para designar un estado y un “1” para el otro. Por ejemplo:

- ¿Ha llovido hoy en la ciudad?: sí (1) / no (0).
- Estado del interruptor de la luz: 0 = apagado, 1 = encendido.
- ¿Cómo está la plaza de garaje en el parking?: 0 = vacía, 1 = ocupada.

¿Te has fijado que en algunos aparcamientos de los centros comerciales encima de cada plaza de aparcamiento hay una lucecita que está verde cuando la plaza está vacía y roja cuando está ocupada?



En este caso el estado de la plaza se está representando con un bit de estado (ocupado/libre) y nosotros lo vemos como una diferencia de color en la luz, pero en el sistema informático que se encuentra controlando la ocupación del parking, el estado de esa plaza se va a representar con un bit, que matemáticamente escribimos como “0” (cero) o “1” (uno).

Bit, Byte y palabra

Sabes que una información que tiene solo dos posibles valores se puede expresar con un bit (0/1) pero, ¿qué ocurre si quiero expresar digitalmente una información más compleja? ¿Alguna que tenga muchos posibles valores, una gran cantidad de información?

Bien, respondiendo a la cuestión anterior lo que podemos hacer es expresarla como una secuencia de bits, cuya combinación represente todos los posibles estados.

Por ejemplo, supongamos que tenemos un ascensor en un edificio de 7 pisos y queremos numerar cada uno de los posibles puntos de parada con una secuencia de bits. En este caso podríamos usar, por ejemplo, la siguiente combinación:

- 111 – piso 7
- 110 – piso 6
- 101 – piso 5
- 100 – piso 4
- 011 – piso 3
- 010 – piso 2
- 001 – piso 1
- 000 – bajo

En el ejemplo anterior puedes ver como con 3 bits (cada uno de ellos pudiendo tomar el valor de "0" o "1") podemos representar una información con 8 estados posibles. Si disponemos de un número mayor de bits podremos representar informaciones más complejas.

En general debes recordar que con un número "**n**" de bits se pueden representar 2^n posibles estados.

Es decir, con 8 bits podemos representar hasta 256 posibles estados diferentes. ¡Por ejemplo en un rascacielos un montón de paradas del ascensor!

Pues precisamente 8 bits es lo que forman una unidad llamada "**byte**", o lo que es lo mismo, **un byte es una secuencia de 8 bits**. Como curiosidad te diremos que al principio el byte se definió de forma más general, y podía ser otra cantidad de bits, pero actualmente está generalizado que:

$$\text{1 byte} = \text{8 bits}$$

A veces también encontrarás el término "**palabra**" para referirnos normalmente a varios bytes que se manejan de forma conjunta, por ejemplo para señalar el contenido de posiciones de memoria, expresar una dirección etc. El tamaño de una palabra era normalmente de **2 bytes**, es decir, una palabra eran **16 bits**, pero al aumentar la potencia de los procesadores modernos podemos tener sistemas que trabajen con un tamaño de palabra de **32 bits** o de **64 bits**, por ejemplo.

Pero, ¿y si tenemos una gran cantidad de información? ¿Y si necesitamos muchísimos bytes para almacenarla?

Bien, al final, cualquier información ocupará un determinado número de bytes y por tanto una determinada cantidad de bits. Tenemos unidades que nos permiten cuantificarla en múltiplos de la unidad base, y que expresamos normalmente con un “prefijo”, como pueden ser: “**kilobits**”, “**kilobytes**”, “**megabits**” o “**megabytes**”, etc. Esto es algo que debes conocer y tener claro. Te mostramos a continuación un cuadro de unidades y múltiplos:

PREFIJO	UNIDAD	SÍMBOLO	VALOR
	byte	B	$10^0 = 1$
kilo	kilobyte	KB	$10^3 = 1\,000$
mega	megabyte	MB	$10^6 = 1\,000\,000$
giga	gigabyte	GB	$10^9 = 1\,000\,000\,000$
tera	terabyte	TB	$10^{12} = 1\,000\,000\,000\,000$
peta	petabyte	PB	$10^{15} = 1\,000\,000\,000\,000\,000$
exa	exabyte	EB	$10^{18} = 1\,000\,000\,000\,000\,000\,000$
zetta	zettabyte	ZB	$10^{21} = 1\,000\,000\,000\,000\,000\,000\,000$
yotta	yottabyte	YB	$10^{24} = 1\,000\,000\,000\,000\,000\,000\,000\,000$

¿Cuándo usar las unidades en “bits” y cuándo en “bytes”?

Lo primero es que te acostumbres a escribir los múltiplos de las unidades. Empleamos las minúsculas cuando hablamos de bits y las mayúsculas cuando hablamos de bytes. Por ejemplo, escribimos:

- 1 Kb = 1 kilobit
- 1 KB = 1 Kilobyte
- 2 Mb = 2 Megabits
- 2 MB = 2 Megabytes
- etc.

Algo importante y que te encontrarás más adelante es que se suele expresar de forma diferente la cantidad de información que almacenamos en un lugar (por ejemplo una memoria) la cual expresamos normalmente en múltiplos de bytes, y la cantidad de información que estamos transfiriendo entre dos máquinas, que se suele expresar en múltiplos de bits.

Así decimos que, por ejemplo, nos hemos comprado un disco duro de **1 TeraByte**, o que tenemos **4 GB de memoria**. Por otro lado podemos tener una línea de transmisión de acceso a Internet de **50 Mbps** (Megabits por segundo). Esto de nuevo es un acuerdo adoptado por la generalidad del mundo técnico, pero muchas veces verás que en la publicidad y la información de venta de los productos no se sigue a rajatabla.

Un ejemplo sencillo

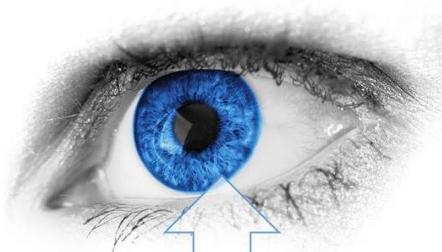
Veamos cómo es un fichero "por dentro". En la figura de abajo puedes ver la imagen de un ojo humano, y cómo podría ser parte del contenido de ese fichero de imagen, todo sería muy largo para ponerlo.

¿Ves algo que te llame la atención?

En efecto, son un montón de “unos” y “ceros” encadenados que parecen no tener un orden ni un formato, pero en realidad cada tipo de archivo (una imagen, un texto, un audio) tiene su propio formato, pero eso lo veremos más adelante.

Lo importante ahora es que recuerdes que la información en los sistemas de información se guarda y trabaja en modo binario.

Como verás también más adelante, **cualquier ordenador solamente entiende de "ceros y unos"** y luego tiene programas que adaptan esa información y nos la presentan de modo que los humanos podamos entenderla; verla si es una imagen, escucharla si se trata de música, etc.



Y además eso lo hacen muy rápido.

Diferentes sistemas de numeración

Para los humanos trabajar con la información en binario es algo muy difícil, pero lo cierto es que algunas veces necesitamos bajar al nivel de ver los bits que hay en una determinada posición de memoria o cuál es el contenido de un programa que se está ejecutando en un procesador.

Normalmente tenemos herramientas que nos facilitan trabajar con la información en un lenguaje más entendible para nosotros, pero, ¿qué hacemos cuando no disponemos de ellas o a pesar de todo tenemos que ver “los bits”?

Para poder trabajar mejor con la información a niveles tan bajos se emplean dos **sistemas de numeración** distintos al sistema binario; nos referimos al **sistema hexadecimal** y al **sistema octal**. Vamos a verlos.

Sistema decimal

Este es el que usamos normalmente para contar, donde tenemos diez dígitos (0, 1, 2, 3, 4, ... 9) y los números se expresan como una secuencia, donde la posición de cada dígito tiene un peso e indica un múltiplo de una potencia de diez. Por ejemplo el número:

$$14_{10} = (1 \times 10^1) + (4 \times 10^0)$$

Hasta aquí es fácil, pero puede ser más complicado:

$$1879'25_{10} = (1 \times 10^3) + (8 \times 10^2) + (7 \times 10^1) + (9 \times 10^0) + (2 \times 10^{-1}) + (5 \times 10^{-2})$$

Sistema binario

Es el que has visto en los ejemplos de digitalización de la información, en el que tenemos solamente dos dígitos (“1” y “0”) y la información (números en este caso) se representa como una secuencia de bits. Al igual que en el decimal, la situación de los dígitos también tiene un peso, pero en este caso se emplean potencias de 2. Por ejemplo:

$$01110_2 = (1 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) \text{ sería igual a } 14 \text{ en decimal.}$$

Como ves, es posible expresar un mismo número tanto en un sistema como en otro. Vamos a ver otros dos y luego practicaremos un poco.

Sistema octal

En este sistema tendremos ocho posibles dígitos: 0, 1, 2, 3, 4, 5, 6, 7 y con ellos expresaremos los números, al igual que en los anteriores, con unos pesos para cada posición, que serán potencias de 8. Veamos un ejemplo:

$$14_8 = (1 \times 8^1) + (4 \times 8^0)$$
 es decir 14_8 es igual a 12_{10} (14 en base ocho es igual a 12 en base diez)

Fíjate que al cambiar el sistema los mismos dígitos representan cantidades diferentes. No te preocupes, no es tan complicado como parece y dentro de poco aprenderás a “traducir” números de un sistema a otro.

Sistema hexadecimal

Por último veremos el sistema de numeración hexadecimal, en el que tendremos dieciséis dígitos para representar los valores de los datos, y te preguntarás, ¿cómo es eso posible si solamente tenemos diez dígitos numéricos?

Bueno, la respuesta es sencilla, además de los números del 0 al 9 utilizaremos seis letras: A, B, C, D, E y F. Es decir, un número en hexadecimal puede tener el aspecto siguiente: 7E2 (que sería igual a 2018 en decimal).

$$7E2_{16} = (7 \times 16^2) + (E \times 16^1) + (2 \times 16^0) = 1792 + (14 \times 16) + (2 \times 1) = 1792 + 224 + 2 = 2018_{10}$$

Aquí puedes ver cómo los números hexadecimales pueden tener “letras” (entre la A y la F) en su composición. En este caso la “E” representaría el 14, que al multiplicarlo por 16 nos da 224 y en la posición del extremo derecho siempre tendremos la potencia de la base numérica elevada a “cero” (y recuerda que cualquier número elevado a cero es igual a 1) por lo cual el dígito que la multiplique siempre será él mismo multiplicado por “1”.

NOTA: en este punto, por si no lo habías deducido ya, te diremos que la base numérica de cada sistema es la que se va elevando a las diferentes potencias en cada posición del número. El sistema hexadecimal tiene por base “16”, el decimal “10”, el octal “8” y el binario “2”.

Además, date cuenta que cuando queremos especificar que un número está expresado en una determinada base numérica la especificamos como un **subíndice** a la derecha de sus dígitos, y no debes confundir esto con la forma en la que se ponen las potencias, que es con un **superíndice**, también a la derecha.

Conversión entre sistemas de numeración

Veamos entonces cómo se convierte un número de un sistema a otro, o lo que es lo mismo, cómo podemos expresar una determinada cantidad en varios sistemas.

Esto debes tenerlo claro, pues será de mucha utilidad más adelante cuando trabajemos con sistemas reales.

Una primera tabla de conversión de los primeros números sería la siguiente:

DECIMAL	BINARIO	OCTAL	HEXADECIMAL
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Pero esto solamente nos vale para los quince primeros números. ¿Cómo hacemos para convertir números más grandes?

Conversión de cualquier sistema a decimal

Para pasar de binario a decimal, de octal a decimal o de hexadecimal a decimal, bastará con hacer la operación de multiplicar cada dígito del número por la potencia correspondiente a la posición en la que se encuentra, sumando el resultado.

Es lo que hemos estado haciendo durante la explicación anterior de cada sistema. Otros ejemplos pueden ser:

$$1457_{10} = (1 \times 1000) + (4 \times 100) + (5 \times 10) + (7 \times 1) \text{ aunque en este caso ya estaba en decimal}$$

$$101_2 = (1 \times 4) + (0 \times 2) + (1 \times 1) = 5_{10}$$

$$5B1_{16} = (5 \times 16^2) + (11 \times 16^1) + (1 \times 16^0) = (5 \times 256) + (11 \times 16) + (1 \times 1) = 1457_{10}$$

Conversión de decimal a binario

En este caso podríamos calcular el número de potencias de 2 cuya suma nos daría el número decimal, pero si tenemos números grandes quizás sea mejor el método de la división sucesiva por 2, que consiste en ir dividiendo sucesivamente por 2 el número original, hasta que el cociente nos dé una parte entera igual a cero, y entonces el último cociente y los restos de las divisiones serán el número en base dos.

Veámoslo mejor con un ejemplo. Nos proponemos pasar el número 113_{10} (es decir "113" en base 10) a binario:



Fíjate que al recopilar los "1" y "0" del último cociente y los restos de las divisiones, tenemos que ponerlos "al revés" para obtener el número en binario, es decir, el bit de mayor peso es el último cociente, y el de menor peso el primer resto de la primera división. Puedes verlo en el ejemplo anterior.

Conversión de binario a hexadecimal

Conversión de binario a hexadecimal

En este caso lo tenemos más sencillo, porque basta con dividir el número binario en grupos de 4 bits (comenzando por la derecha) y luego reemplazar cada grupo de 4 bits por su dígito en hexadecimal correspondiente. Veamos un ejemplo:

$$110000011010011101011110_2 = 1100\ 0001\ 1010\ 0111\ 0101\ 1110 = \text{C1A75E}_{16}$$

Conversión de hexadecimal a binario

Todavía más sencillo que el anterior, pues basta sustituir cada dígito hexadecimal por su equivalente en binario. Algunos ejemplos:

$$\mathbf{10AF_{16}} = 0001\ 0000\ 1010\ 1111 = \mathbf{0001000010101111_2}$$

$$\mathbf{CF7E_{16}} = 1100\ 1111\ 0111\ 1110 = \mathbf{1100111101111110_2}$$

$$\mathbf{5742_{16}} = 0101\ 0111\ 0100\ 0010 = \mathbf{0101011101000010_2}$$

Conversión de binario a octal

Si tenemos un número en binario y queremos convertirlo a base ocho, haremos lo mismo que para pasarlo a hexadecimal, pero aquí los grupos en los que dividiremos el número serán de tres bits. Por ejemplo:

$$\mathbf{110000011010011101011110_2} = 110\ 000\ 011\ 010\ 011\ 101\ 011\ 110 = \mathbf{60323536_8}$$

Conversión de octal a binario

Aquí también lo tenemos fácil, porque basta con sustituir cada dígito en base ocho por su equivalente binario. Por ejemplo:

$$\mathbf{76432_8} = 111\ 110\ 100\ 011\ 010 = \mathbf{111110100011010_2}$$

Nota: todo esto es más complicado cuando trabajamos con números negativos y fracciones, pero de momento no vamos a profundizar más porque excede el objetivo de esta lección.

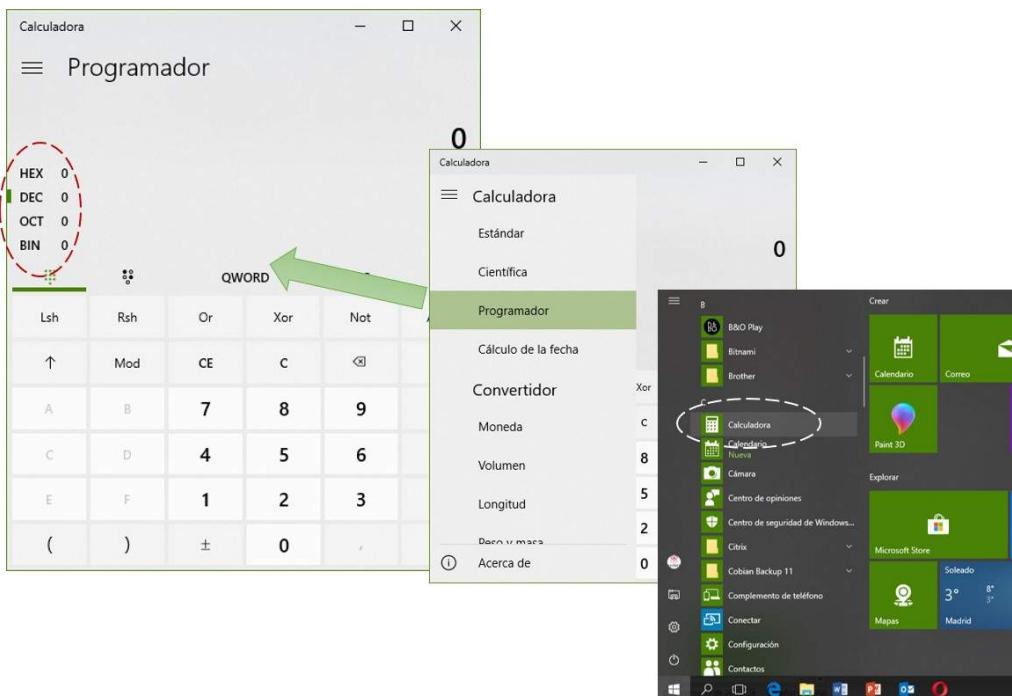
Ejercicio a resolver: conversión de números

Afortunadamente también existen multitud de programas que nos ayudan a hacer la conversión entre diferentes sistemas, empezando por la calculadora del propio Windows.

Como ejercicio te proponemos que practiques traduciendo a **hexadecimal**, **octal** y **binario** los siguientes números que están en **decimal**:

- 2017
- 1961
- 4545
- 1000
- 1024
- 2048
- 4096
- 0256
- 0512

Puedes usar la calculadora de Windows o bien cualquier otro programa de ayuda de los muchos que hay en Internet, pero te recomendamos que primero lo hagas “a mano”, solamente con el lápiz y tu cabeza, y luego lo compruebes con el programa.



Si usas la calculadora de Windows te recomendamos que la pongas en "**modo programador**" para que te ofrezca las opciones de conversión directa entre sistemas numéricos.

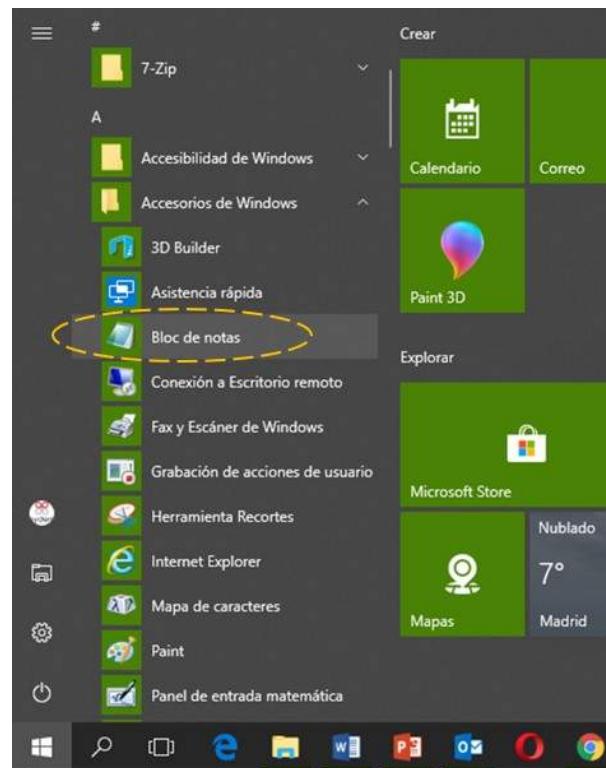
Ejercicio

Abre un fichero de texto con el bloc de notas y escribe las siguientes combinaciones de caracteres "ALT+128" y "ALT+0128". Después de hacerlo observa:

¿Han generado las mismas combinaciones de caracteres?

Verás que no es así, una genera el carácter "Ç" y la otra el "€". ¿Por qué ocurre esto?

El uso de la tecla "ALT izquierda" + "otra tecla" o una "combinación" es lo que llamamos un "**atajo de teclado**", muy usado en Windows. Se puede usar un atajo de teclado para insertar caracteres que no están disponibles en el teclado del ordenador.

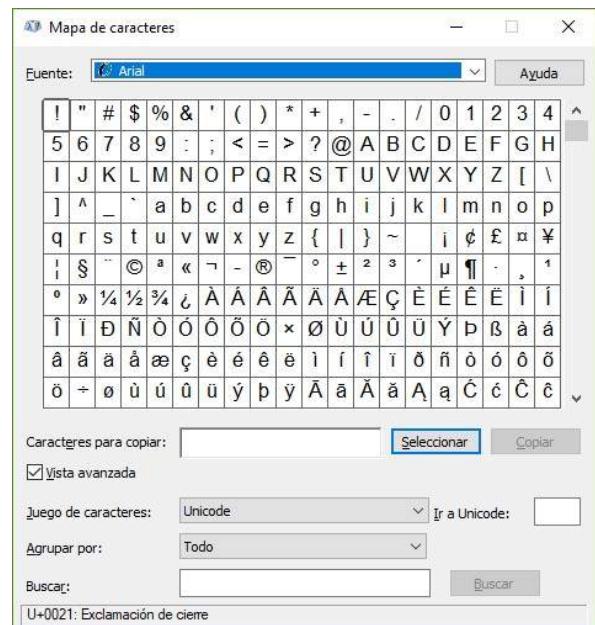


Si pulsamos solamente "ALT+0128" estamos usando un juego de "**caracteres OEM** (Original Equipment Manufacturer)" que el fabricante incluye en el sistema operativo, y que depende además de la ubicación geográfica. En cambio, si pulsamos "ALT+128" estamos usando el **juego de caracteres ASCII Extendido**, que puedes consultar en muchos sitios, por ejemplo:
<http://www.asciiitable.com/>

El **Mapa de caracteres** de Windows.

Además de todo lo anterior, en Windows puedes activar un pequeño programa llamado "Mapa de caracteres" (puedes insertarlo en el espacio de búsqueda al lado de "Inicio" y el sistema lo encontrará) con el que podemos visualizar los conjuntos de caracteres asociados a una determinada fuente o tipo de letra. En la figura puedes ver, por ejemplo, los caracteres Unicode asociados a la fuente "Arial".

¡Ojo! porque NO coinciden con los que vimos antes; prueba a buscar el que tiene el código 128 ;-).



Codificación de la información

Hasta ahora hemos visto cómo representar números en varios sistemas y cómo se pueden expresar en un sistema binario, que es el que entienden los ordenadores, pero, ¿qué ocurre cuando queremos representar un texto? ¿Y un fichero de código ejecutable? ¿Cómo lo conseguimos?

Los datos que manejan los ordenadores pueden ser de muchos tipos: información numérica, texto, imágenes fijas o en movimiento, audio, instrucciones de un programa a ejecutar, etc. Todos ellos han de expresarse en formato binario, que es lo que entienden los procesadores. Para ello se sigue un proceso de “**codificación**”, es decir, se generan secuencias de bits de acuerdo a unas normas (código) para representar la información de origen.

Existen muchísimos tipos de códigos para representar y trabajar con la información digital binaria, algunos específicamente para codificar texto por ejemplo, otros para generar ficheros ejecutables, etc. Nosotros vamos a ver, a modo de ejemplo, algunos de los códigos más utilizados.

El código ASCII (American Standard Code for Information Interchange)

El código ASCII es muy utilizado en comunicaciones de datos. Utilizaba en principio solo 7 bits para representar 128 posibles combinaciones, cada una de ellas la asociada a un carácter o símbolo. Más adelante se creó el “Código ASCII Extendido” con 256 combinaciones.

CARACTERES ASCII DE CONTROL		
00	NULL	(carácter nulo)
01	SOH	(inicio encabezado)
02	STX	(inicio texto)
03	ETX	(fin de texto)
04	EOT	(fin transmisión)
05	ENQ	(consulta)
06	ACK	(reconocimiento)
07	BEL	(timbre)
08	BS	(retroceso)
09	HT	(tab horizontal)
10	LF	(nueva línea)
11	VT	(tab vertical)
12	FF	(nueva página)
13	CR	(retorno de carro)
14	SO	(desplaza afuera)
15	SI	(desplaza adentro)
16	DLE	(esc. vínculo datos)
17	DC1	(control disp. 1)
18	DC2	(control disp. 2)
19	DC3	(control disp. 3)
20	DC4	(control disp. 4)
21	NAK	(conf. negativa)
22	SYN	(inactividad sinc.)
23	ETB	(fin bloque trans)
24	CAN	(cancelar)
25	EM	(fin del medio)
26	SUB	(sustitución)
27	ESC	(escape)
28	FS	(sep. archivos)
29	GS	(sep. grupos)
30	RS	(sep. registros)
31	US	(sep. unidades)
127	DEL	(suprimir)

CARACTERES ASCII IMPRIMIBLES		
32	espacio	64 @
33	!	65 A
34	"	66 B
35	#	67 C
36	\$	68 D
37	%	69 E
38	&	70 F
39	'	71 G
40	(72 H
41)	73 I
42	*	74 J
43	+	75 K
44	,	76 L
45	-	77 M
46	.	78 N
47	/	79 O
48	0	80 P
49	1	81 Q
50	2	82 R
51	3	83 S
52	4	84 T
53	5	85 U
54	6	86 V
55	7	87 W
56	8	88 X
57	9	89 Y
58	:	90 Z
59	:	91 [
60	<	92 \
61	=	93]
62	>	94 ^
63	?	95 _

ASCII EXTENDIDO (PÁGINA DE CÓDIGO 437)					
128	ç	160 á	192 l	224 ó	
129	ú	161 í	193 ñ	225 þ	
130	é	162 ó	194 t	226 ô	
131	â	163 ú	195 ã	227 ò	
132	ã	164 ñ	196 —	228 õ	
133	à	165 Ñ	197 þ	229 ö	
134	å	166 ø	198 ã	230 µ	
135	§	167 ø	199 Ä	231 þ	
136	è	168 Ł	200 ll	232 þ	
137	ë	169 Ø	201 Þ	233 Ú	
138	è	170 ñ	202 ï	234 Ù	
139	Í	171 ½	203 ¶	235 Ú	
140	í	172 ¼	204 ¶	236 Ú	
141	ì	173 ï	205 —	237 Ý	
142	Ã	174 «	206 ¶	238 —	
143	À	175 »	207 »	239 —	
144	É	176 ☁	208 ð	240 —	
145	æ	177 ☀	209 Đ	241 ±	
146	Æ	178 ☀	210 È	242 =	
147	ô	179	211 È	243 %	
148	ö	180 ï	212 È	244 ¶	
149	ò	181 Á	213 i	245 §	
150	û	182 Ä	214 ï	246 ÷	
151	ù	183 À	215 ï	247 ,	
152	ğ	184 Ø	216 ■	248 ø	
153	ö	185 ¶	217 J	249 —	
154	Ù	186 ¶	218 r	250 —	
155	ø	187 ¶	219 ■	251 :	
156	£	188 ¶	220 ■	252 :	
157	Ø	189 ¢	221 :	253 :	
158	¤	190 ¥	222 ï	254 ■	
159	ƒ	191 ¢	223 ■	255 nbsp	

Puedes encontrar la lista entera de caracteres en <https://es.wikipedia.org/wiki/ASCII>, pero no se trata de que los aprendas, simplemente recuerda que puedes usarlos e insertarlos en cualquier archivo de texto usando el teclado del ordenador y pulsando la tecla "ALT" mientras escribes el código del carácter. Por ejemplo, para insertar el carácter del euro pulsaremos "ALT+0128". Haz la prueba.

El código EBCDIC (Extended Binary Coded Decimal Interchange Code)

Es otro código muy utilizado, creado por IBM para sus ordenadores mainframe. Utiliza 8 bits y dispone de 256 combinaciones de caracteres alfanuméricos, símbolos y controles.

Código BCD (Binary-Coded Decimal)

Es un código para representar números decimales en modo binario. En este caso no se hace la transformación simple de base de numeración, sino que tenemos que hacer la conversión según el código indica. Utilizando el código BCD cada dígito en decimal se representa con una secuencia independiente de 4 bits, es decir, por cada dígito en decimal que tenga nuestro número necesitaremos cuatro bits para codificarlo. Para los dígitos del 0 al 9 tendríamos la siguiente codificación:

Decimal:	0	1	2	3	4	5	6	7	8	9
BCD:	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

De esta forma, para el mismo ejemplo anterior, en el que convertíamos el número **113** (decimal) a binario y nos daba como resultado "**1110001**", si codificamos el mismo número siguiendo el código BCD y sustituyendo cada dígito por sus respectivos 4 bits obtendríamos:

113₁₀ al pasarlo a BCD nos daría: **0001 0001 0011_{BCD}**

Como vemos, el resultado de pasar 113 a binario directamente (000001110001) o ponerlo en código BCD (000100010011) no es el mismo.

DECIMAL	NATURAL	AIKEN	EXCESO 3
0	0000	0000	0011
1	0001	0001	0100
2	0010	0010	0101
3	0011	0011	0110
4	0100	0100	0111
5	0101	1011	1000
6	0110	1100	1001
7	0111	1101	1010
8	1000	1110	1011
9	1001	1111	1100

Date cuenta también que, para comparar con el mismo número de bits ambos resultados, hemos rellenado con ceros por la izquierda el primer caso.

Por si esto fuera poco, te diremos que existen diferentes tipos de códigos BCD, y aunque todos utilizan 4 bits, las combinaciones correspondientes a los dígitos en decimal no son las mismas. En la figura tienes algunos ejemplos.

Ejercicio a resolver: codificación

Tomando como ejemplo la tabla de caracteres ASCII de la figura (con 7 bits), intenta escribir en binario y hexadecimal el siguiente texto:

En un lugar de la mancha de cuyo nombre no quiero acordarme

BITS				b ₇	b ₆	b ₅	0	0	0	0	0	1	0	1	0	0	1	1	0	0	1	1	1	0	1	1	1
b ⁴	b ³	b ²	b ¹	COLUMN				ROW	0	1	2	3	4	5	6	7											
0	0	0	0	0				NUL	DLE	SP	0	@	P	\	p												
0	0	0	1	1				SOH	DC1	!	1	A	Q	a	q												
0	0	1	0	2				STX	DC2	"	2	B	R	b	r												
0	0	1	1	3				ETX	DC3	#	3	C	S	c	s												
0	1	0	0	4				EOT	DC4	\$	4	D	T	d	t												
0	1	0	1	5				ENQ	NAK	%	5	E	U	e	u												
0	1	1	0	6				ACK	SYN	&	6	F	V	f	v												
0	1	1	1	7				BEL	ETB	/	7	G	W	g	w												
1	0	0	0	8				BS	CAN	(8	H	X	h	x												
1	0	0	1	9				HT	EM)	9	I	Y	i	y												
1	0	1	0	A				LF	SUB	=	:	J	Z	j	z												
1	0	1	1	B				VT	ESC	+	;	K	[k	{												
1	1	0	0	C				FF	FS	,	<	L	\	l													
1	1	0	1	D				CR	GS	-	=	M]	m	}												
1	1	1	0	E				SO	RS	.	>	N	^	n	~												
1	1	1	1	F				SI	US	/	?	O	-	o	DEL												

Te ayudamos con el principio:

HEX	BINARIO
E = 45	1000101
n = 6E	1101110
= 7F	0100000 (el espacio es un carácter)
u = 75	1110101
n = 6E	
= 7F	
l = sigue tú.	

¿Qué son los protocolos de datos?

Cuando un sistema informático quiere intercambiar datos con otro sistema tenemos que disponer de algún medio de comunicación entre ambos.

En un esquema sencillo podríamos representar el proceso en el diagrama:



En este caso está representado un solo sentido de envío de información, pero si tenemos una comunicación bidireccional puedes imaginarte que el mismo esquema es válido en sentido contrario, y el lado que ahora es destino puede funcionar como origen de información y viceversa.

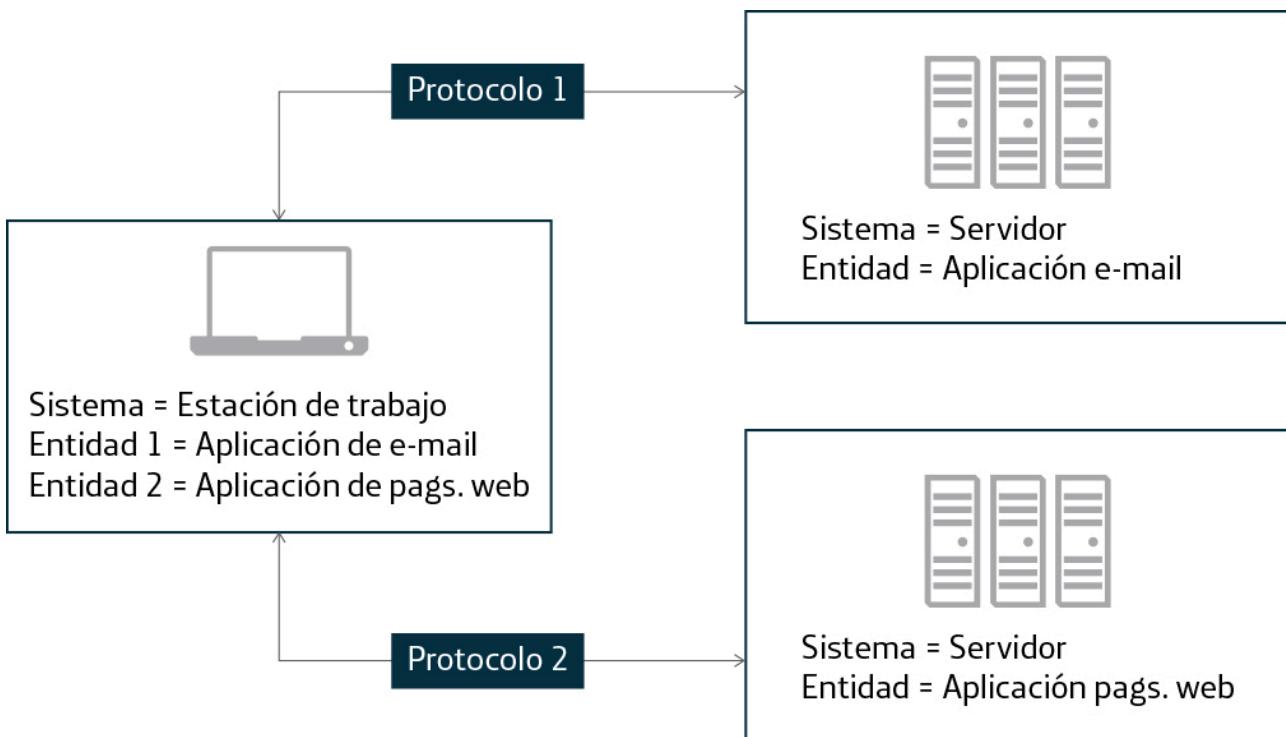
Por ejemplo, cuando enviamos un correo electrónico desde nuestro ordenador estamos funcionando como origen y transmisor de información, pero también como receptor, puesto que el sistema de correo recibe una confirmación de que el envío se ha hecho correctamente y esa es una información que nos viene en sentido contrario.

El objetivo en cualquier intercambio de datos es que la información de origen llegue al destino en perfectas condiciones, sin errores ni pérdida de bits por el camino. Para lograrlo establecemos un procedimiento de intercambio con una serie de normas. Esto es lo que llamamos "**protocolo de transmisión o intercambio de datos**".

¿Qué incluye un protocolo de datos?

Como puedes imaginar, hay muchos protocolos de datos y estos van también evolucionando con el tiempo, pero en general podemos decir que un protocolo de datos va a especificar:

1. **Una sintaxis:** es decir, una forma en la que agrupar las unidades de datos a enviar, qué información de control añadir a la información útil del usuario, cómo identificar los puntos de origen y destino (por ejemplo con direcciones numéricas), etc.
2. **Una secuencia de diálogo:** indicando quién comienza a emitir y cómo avisar al otro destinatario de que vamos a enviar información, quién debe esperar, cómo reconocer la recepción, cómo contestar a los envíos, qué tipos de envíos puedo usar en cada caso, etc.
3. **Información de gestión y control:** a menudo los protocolos también permiten que los propios sistemas intercambien información sobre ellos mismos, por ejemplo para indicar que están disponibles o bien que no pueden recibir información por un fallo, etc.



Por otro lado, en función de la aplicación o utilidad que vayamos a utilizar, dispondremos de diferentes protocolos y en un mismo sistema puede haber “entidades” diferentes que utilicen cada una un protocolo diferente, o una parte en común y otra no. Por ejemplo, para transmitir un email o visitar una página web utilizamos diferentes aplicaciones (el gestor de correo y el navegador), y ambas utilizan los protocolos de intercambio de redes TCP/IP (Internet es una de ellas), pero luego utilizan protocolos diferentes para enviar un correo electrónico o descargar una página web.

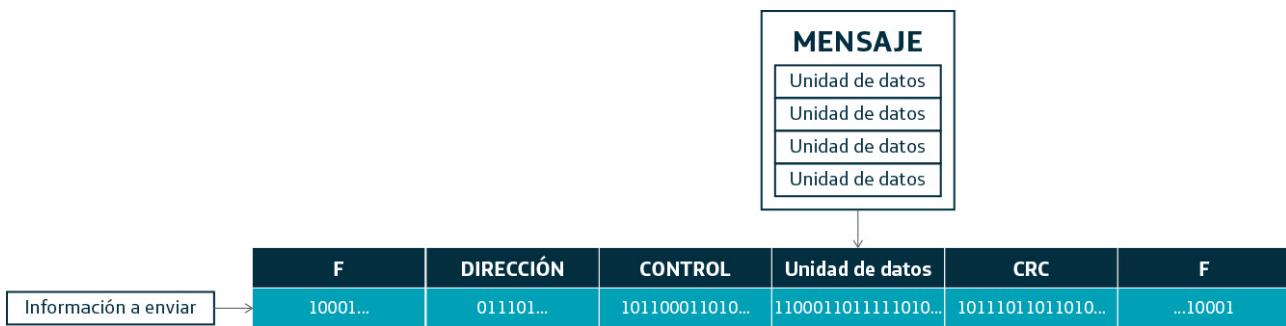
¿Qué es una "trama de información"?

Ya sabes que un protocolo de comunicaciones nos dicta las reglas que hay que seguir para intercambiar información con otros equipos a través de un medio de transmisión o una red.

Dentro de estas normas están especificadas cómo proteger la información a enviar (mensaje). Para ello se suelen hacer varias cosas:

1. Dividir la información en trozos (unidades de datos) a enviar sucesivamente.
2. Proteger cada envío de una unidad de datos con información que intente asegurar que llega bien al destino.
3. Reensamblar el mensaje en destino para poder entenderlo.

Pues bien, a cada una de las unidades que enviamos, y que contienen una unidad de datos con información de gestión del protocolo, es a lo que llamamos “**TRAMA de INFORMACIÓN**”.



Fíjate que normalmente un mensaje se enviará a través de muchas tramas de información. La información que añade el protocolo suele ser, en general, la siguiente:

- **F** = banderas de delimitación (“flags”), indicando el principio y el final de la trama.
- **Dirección** = información de las direcciones de origen y destino a la que tiene que llegar la información.
- **Control** = numeración de la trama para luego ordenarla u otra información de control del protocolo.
- **Unidad de datos** = la parte del mensaje que está siendo enviada y que es realmente lo interesante para el destino.
- **CRC** = códigos de comprobación para detectar posibles errores durante la transmisión.

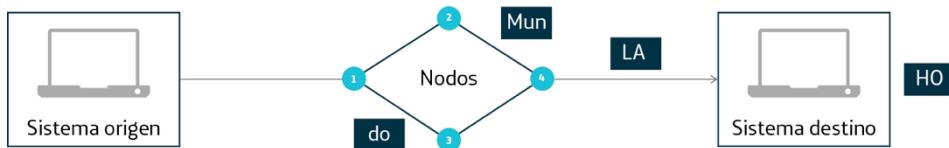
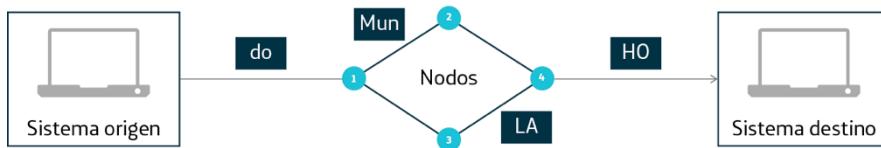
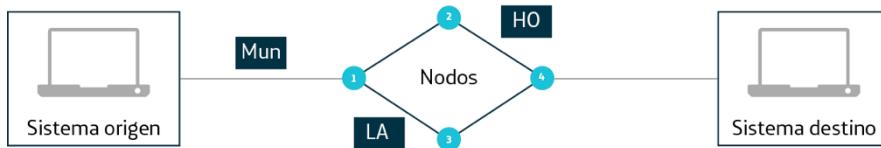
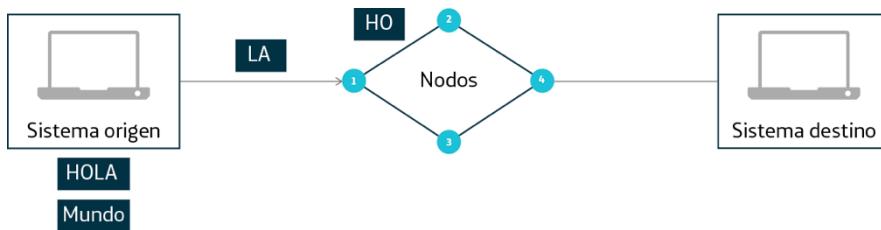
¿Qué es una red de datos?

Normalmente los ordenadores intercambian información entre sí, unos con otros, a través de redes de comunicaciones.

Aunque hay muchos tipos de redes, algunas muy especializadas en un tipo de utilización, en este caso hablaremos en general de las “**redes de datos**”, es decir, redes que están especializadas en el intercambio de datos entre sistemas informáticos.

A menudo también escucharás hablar de las “**redes de paquetes**”, que son redes de datos que intercambian la información digital de los mensajes a transmitir dividiéndola en “trozos” de información y enviándolos hacia el destino utilizando un determinado conjunto de protocolos.

La red está formada por un conjunto de equipos interconectados (“**nodos**”) que reciben la información del origen y se la van pasando de unos a otros hasta entregarla al destino. A veces el camino que siguen los trozos (paquetes) de información no es el mismo para todos ellos. Por eso será necesario controlar cómo llegan y si es necesario reordenarlos al tener todo el mensaje en destino.



En la figura puedes ver un ejemplo sencillo de transmisión del mensaje "Hola mundo" dividido en paquetes de información. Todos ellos irán dentro de tramas que los protegen y seguirán el protocolo de intercambio de la red a la que estuviesesemos conectados.

Por el momento es suficiente con que te quedes con los conceptos básicos. Más adelante habrá lecciones donde profundizaremos en redes y protocolos, fundamentalmente en los protocolos TCP/IP, que son los utilizados en Internet.

Diferencia entre hardware, firmware y software

Trabajando con sistemas informáticos a menudo utilizamos estos términos, así que vamos a repasar su significado antes de seguir adelante.

Para nosotros un sistema informático será cualquier ordenador, desde uno de sobremesa o un portátil hasta un gran ordenador de un centro de cálculo. Es cierto que existen otros sistemas informáticos más complejos y especializados, pero quedan fuera del alcance de este curso.

Cualquier sistema informático (ordenador) tiene una parte física compuesta por la carcasa y todos los circuitos y componentes que tiene dentro, además de los componentes externos que le conectemos.

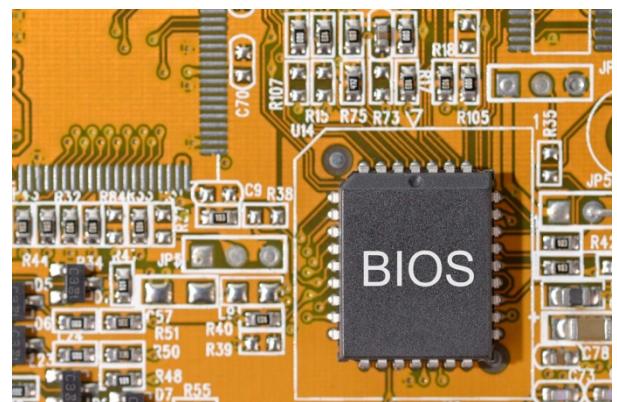
A esto es a lo que llamamos “**hardware (HW)**” (la parte “dura”). Para resumir podemos decir que hardware es todo lo que podemos tocar.



Pero todo este hardware necesita para funcionar una “inteligencia” que lo gobierne y le dote de funcionalidad. Al conjunto de programas, instrucciones, aplicaciones, etc. que montamos sobre el HW y que nos permiten usarlo es a lo que llamamos “**software (SW)**”. Normalmente lo asociamos a todos los programas que “corren” (se ejecutan) sobre el procesador. El SW no podemos “tocarlo”, sino que interactuamos con él a través del HW y las interfaces “hombre-máquina” que tengamos disponibles; los más comunes y conocidos por todos son la pantalla (monitor) del ordenador, el teclado, el ratón, etc. Más adelante verás que a estos elementos les llamamos “periféricos”.



¿Y qué es entonces el “**firmware** (FW)? Pues verás, a veces dentro de un determinado circuito integrado (un chip) insertamos al fabricarlo un determinado programa o una serie de datos. El resultado es un chip HW pero con cierto SW dentro de él. A ese contenido que va grabado dentro del circuito y que normalmente no podemos modificar, le llamamos “firmware” (FW) para distinguirlo del SW que se carga a posteriori, como puede ser un sistema operativo o las aplicaciones de usuario. El ejemplo más conocido quizás sea la BIOS del ordenador, que es un circuito con un programa que se ejecuta cuando encendemos el equipo y se encarga, entre otras cosas, de cargar el SW necesario para su arranque y funcionamiento (el sistema operativo).



Ordenadores. Funcionamiento básico

Aunque más adelante entraremos en mayor profundidad a estudiar la estructura de un sistema informático, vamos a repasar aquí algunos conceptos muy básicos pero esenciales.

Al seguir este curso estarás posiblemente delante de un ordenador (aunque no es la única opción, lo sabemos) y tendrás ante ti una pantalla (monitor), un teclado, un ratón y si es un PC de sobremesa tendrá una unidad central, a la que llamamos a menudo “CPU”, procesador o unidad de proceso. En el caso de que tengas un ordenador portátil los elementos son básicamente los mismos, pero integrados dentro de un solo aparato.

Bien, lo primero a señalar es que la unidad central del ordenador de sobremesa, esa torre más o menos grande a la que conectamos todos los demás elementos, no solamente tiene en su interior una **CPU** (“Central Processor Unit”, un circuito procesador que ejecuta programas) sino muchas más cosas. Por ejemplo, como verás más adelante, una CPU necesita disponer de una memoria principal (**RAM**), y tiene que poder acceder a un **“disco duro”**, o conectarse a través de una circuitería determinada a una **red de cable o inalámbrica (Wifi)**. Todas estas cosas suelen estar dentro de la torre de la unidad central, y en el caso de que tengas un ordenador portátil estarán en el interior del mismo.



Además, externamente solemos tener un **ratón**, un **teclado** y un **monitor** a través de los cuales interactuamos con la máquina (en el PC portátil también están integrados, claro).

En las siguientes lecciones vamos a ver en profundidad el funcionamiento de todo este sistema, con especial atención al elemento que soporta todo su funcionamiento: **el sistema operativo**.

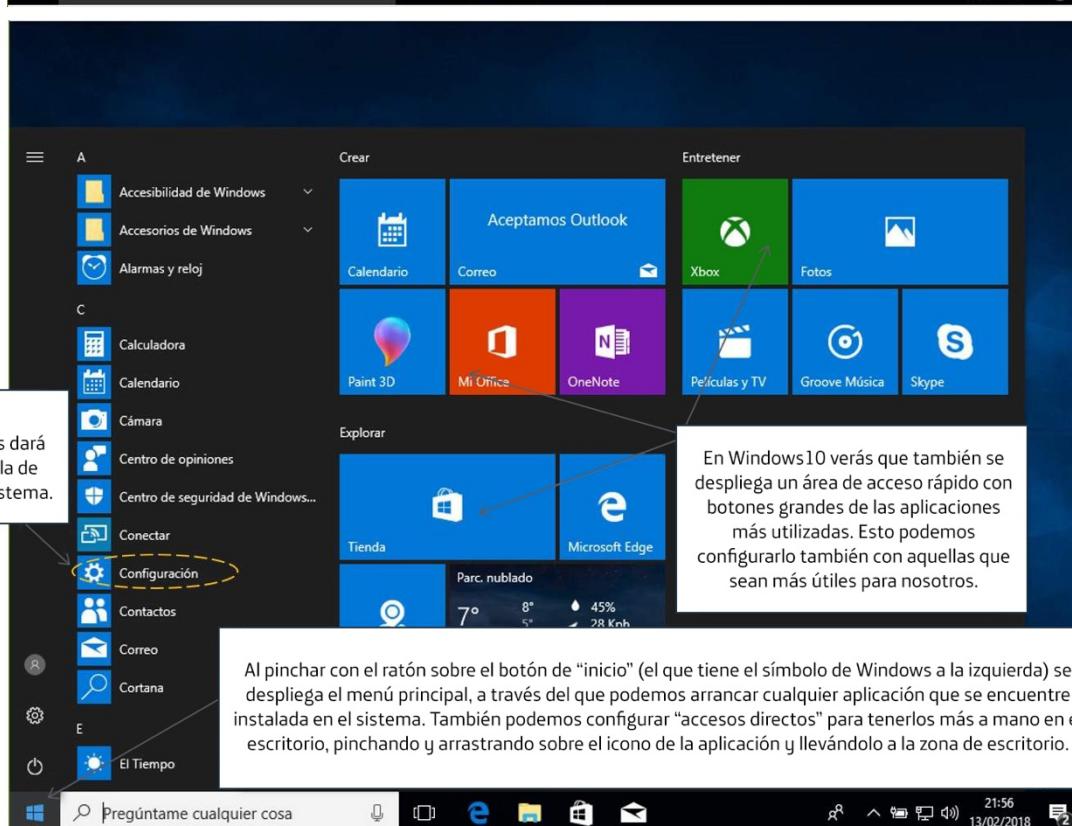
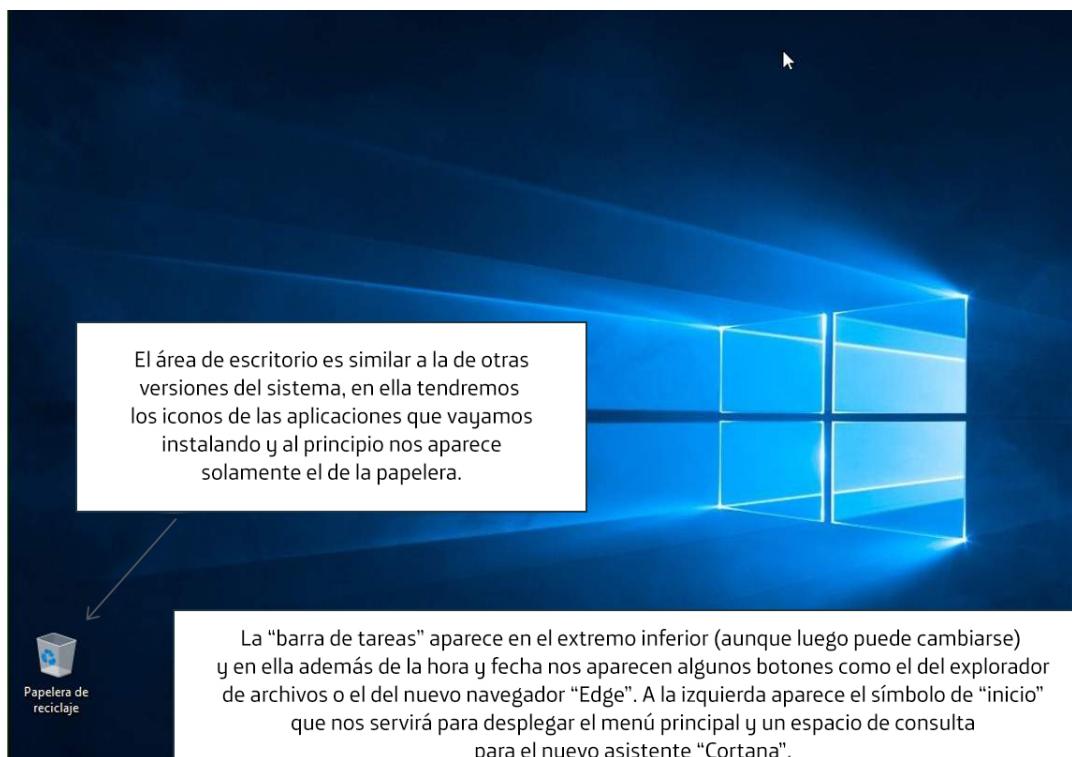
Un vistazo al entorno Windows 10

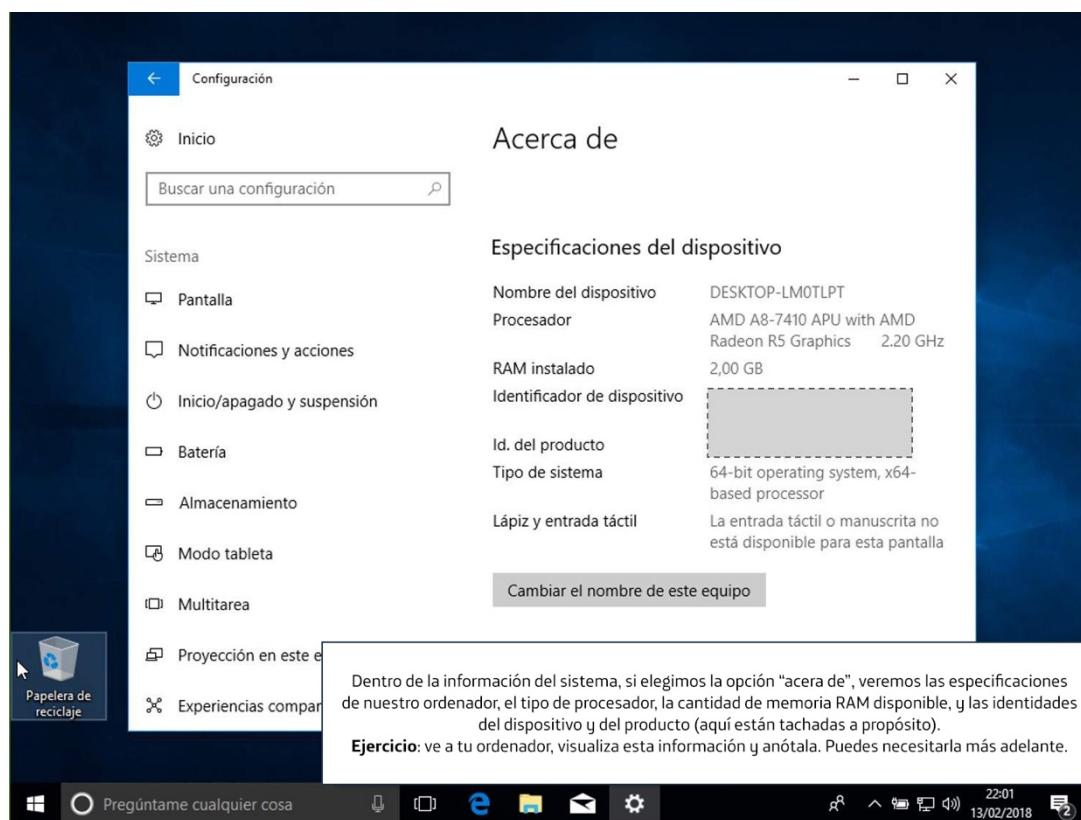
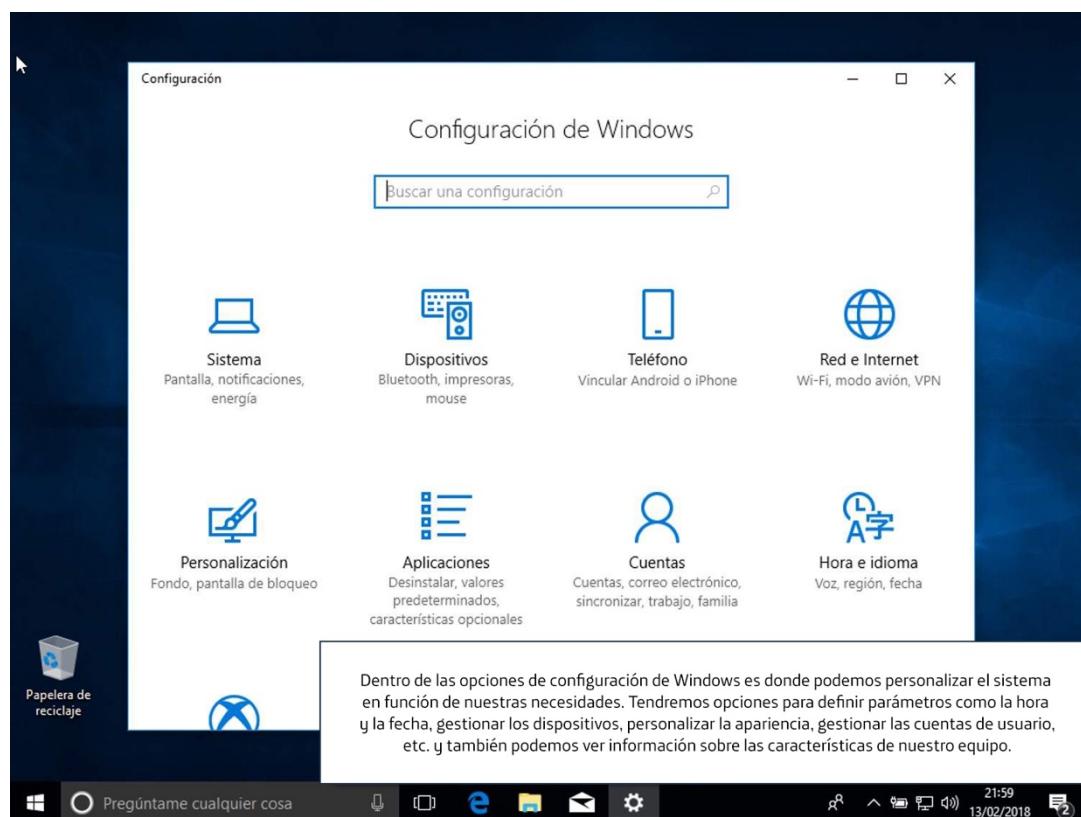
Antes de terminar la lección vamos a echar un vistazo al entorno de Windows 10.

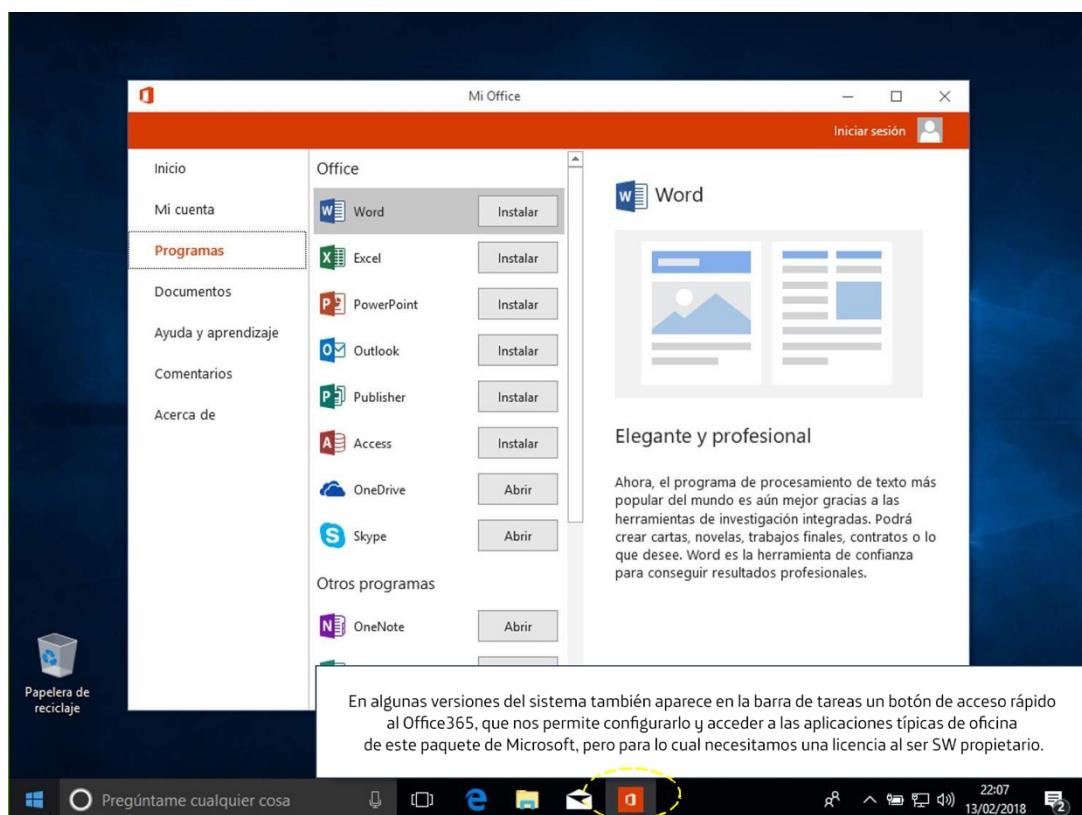
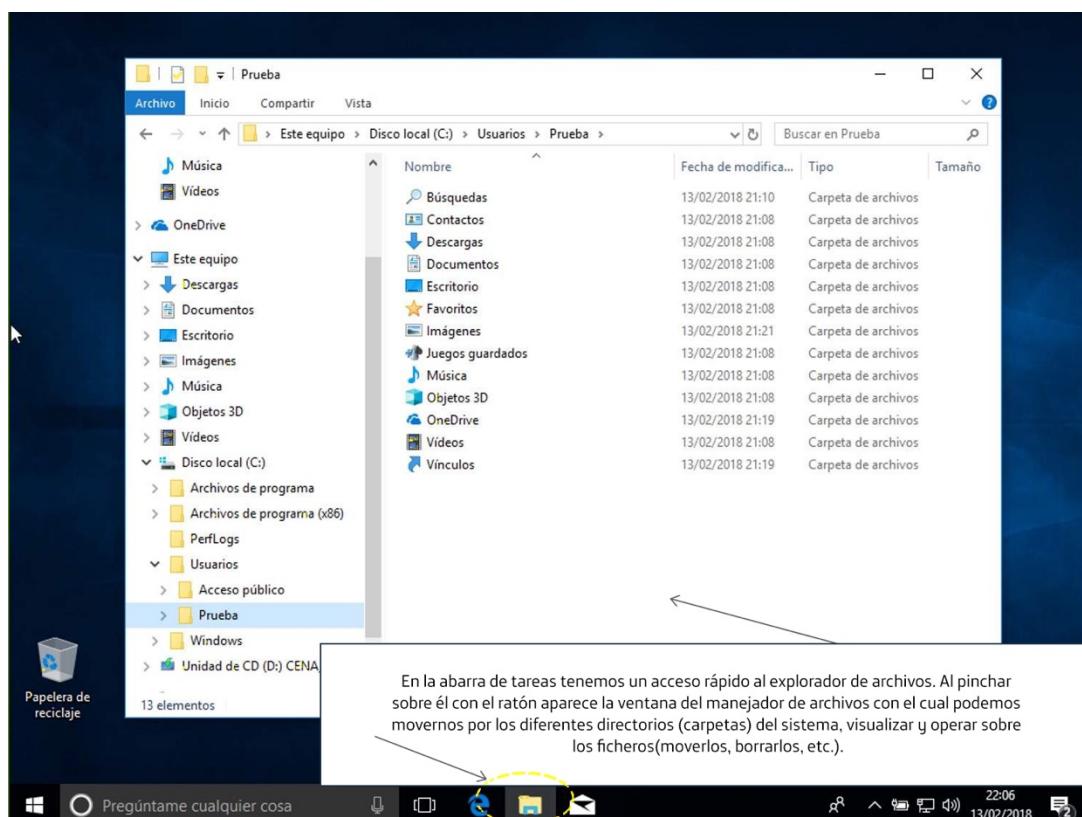
En este curso trabajaremos con ejemplos de varios sistemas operativos, y uno de ellos es Windows 10 de Microsoft.

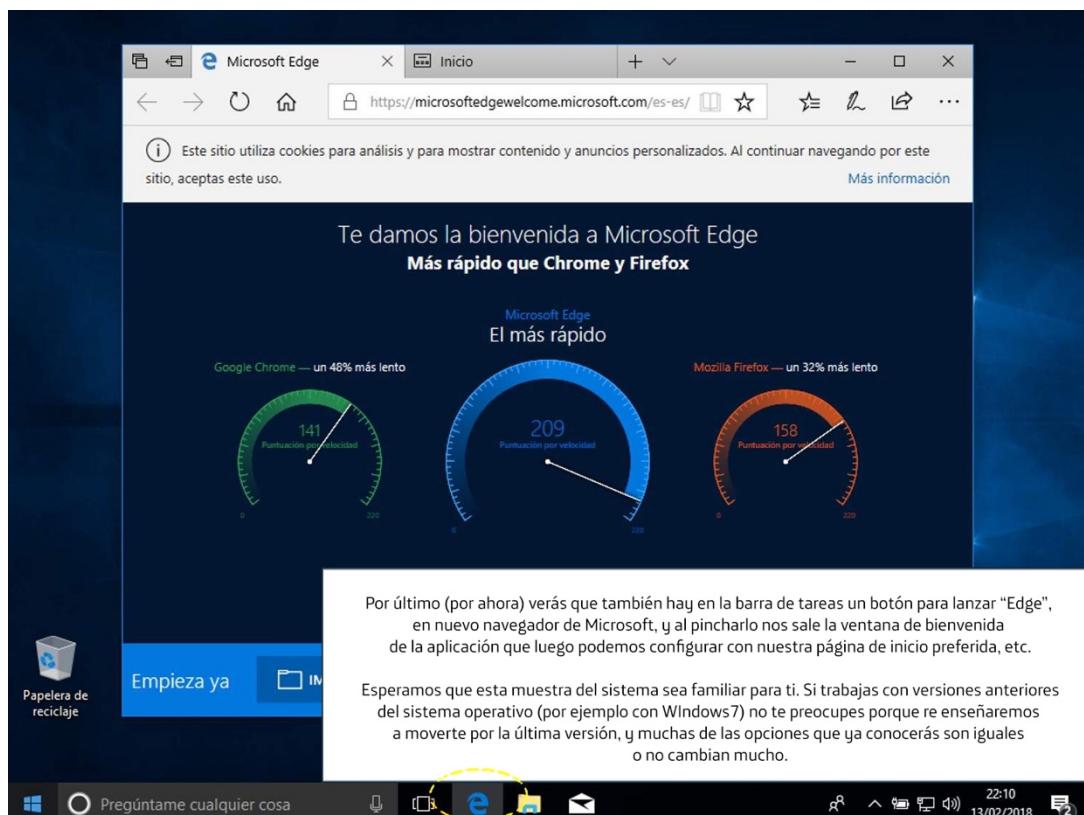
Aunque posiblemente ya conozcas su interfaz de escritorio, vamos a dar un pequeño repaso por si no estuvieras familiarizado con ella. Funciona de forma similar a otras versiones anteriores de Windows, con algunas diferencias de apariencia.

En las figuras siguientes puedes ver el escritorio inicial del sistema operativo, después de una instalación y antes de personalizarlo e instalar aplicaciones, y una breve descripción de sus elementos básicos. Más adelante profundizaremos en su funcionamiento y aprenderás a "sacarle jugo" al sistema.









Despedida

Resumen

Has finalizado esta unidad, veamos los aspectos más importantes que hemos tratado.

Recuerda que en esta lección hemos visto algunos conceptos muy básicos pero muy importantes, como por ejemplo:

- Podemos distinguir entre información analógica e información digital, y ésta última la podemos representar fácilmente con bits de información.
- La cantidad de información que necesitamos podemos expresarla en bits o Bytes y a su vez en múltiplos de estas unidades (Kb, KB, Mb, MBytes, etc.)
- A la hora de representar un número podemos hacerlo en diferentes sistemas, y los más usados son: decimal, binario, octal y hexadecimal.
- También podemos codificar la información, utilizando un código que asocie a cada símbolo o carácter una determinada combinación de bits.
- Saber manejar información binaria y digital en general es importante porque es el lenguaje que se utiliza en los sistemas informáticos y en las comunicaciones de datos.