DATOS E INFORMACIÓN

Conviene diferenciar el significado de los términos **datos** e **información**. La **información** es un concepto muy amplio, que engloba todas las posibles formas del conocimiento humano. Todos los fenómenos que percibimos, ya sea con nuestros sentidos o mediante instrumentos, pueden contener información. Dicha información sólo será inteligible para aquellas personas que conozcan su significado, es decir, para aquellas personas que pueden asociar el significante y su contenido.

Los **datos**, en cambio, son fragmentos de información codificada, lista para ser interpretada y procesada, ya sea por una máquina o por un ser humano. La ventaja del uso de ordenadores estriba en que son capaces de procesar gigantescas cantidades de datos en muy poco tiempo.

Los **datos**, como tales, carecen de **significado**, y solo lo alcanzan cuando son **interpretados**; una vez que los datos han sido procesados y se muestra su resultado de modo inteligible, pasan a formar parte del flujo de información. Los ordenadores pueden almacenar y manipular datos, pero no pueden interpretarlos.

CODIFICACIÓN BINARIA

Para que los ordenadores puedan manipular datos, deben recibirlos codificados. Aunque pueden utilizarse códigos muy diversos, todos los códigos empleados en computación tienen una característica común: sólo utilizan dos signos, los dígitos 0 y 1.

La razón de utilizar sólo dos dígitos se debe a que todos los dispositivos de un ordenador (el procesador, la memoria, etc.) están construidos con circuitos electrónicos basados en transistores, que sólo utilizan dos estados[1]: tensión alta o tensión baja, circuito abierto o circuito cerrado, pasa corriente o no pasa corriente, etc. Asociamos esos estados con los dígitos 1 y 0 y eso nos permite codificar la información.

La codificación binaria está basada en el sistema de numeración binario, que utiliza los dígitos 0 y 1 para representar cualquier número.

UNIDADES DE MEDIDA DE LA INFORMACIÓN

La unidad más pequeña de información es la que corresponde a un suceso en el que sólo hay dos alternativas posibles[2]. Puede representarse con un único dígito binario[3], es decir un 0 o un 1. A este dígito se le denomina ***bit***, abreviatura de la palabra inglesa *binary digit.*

Al conjunto de 8 bits se le denomina ***byte.*** Asimismo, un byte está compuesto por dos **nibble**. Cada **nibble** está compuesto por cuatro bit y se puede representar por un carácter hexadecimal. Por ejemplo, el byte 10100111 está compuesto por dos nibble: 1010 (A) y 0111 (7). Ese byte se representa, en código hexadecimal así: A7

Tanto el bit como el byte son unidades de medida muy pequeñas, por lo que se necesitan algunos múltiplos del byte. Así, hablamos de Kilobyte, Megabyte, Gigabyte, etc. En la tabla siguiente encontrarás la relación entre las distintas magnitudes:

---

[1] Los transistores que componen los circuitos digitales trabajan tan sólo en dos estados: corte (la corriente de colector es nula y la tensión del colector es la misma que la alimentación, unos 5 voltios en tecnología TTL) y saturación (la corriente del colector alcanza el máximo valor posible y la tensión del colector es prácticamente 0 voltios). Así pues, el colector de los transistores sólo conoce dos estados: 5 voltios y 0 voltios.

[2] Cierto o falso. Blanco o negro. Si o no. Etc.

[3] 1 o 0, indistintamente. Es decir, que puede utilizarse una lógica positiva o negativa.

1 Kilobyte (KB)	1024 bytes	8192 bit
1 Megabyte (MB)	1024 Kilobytes	8388608 bit
1 Gigabyte (GB)	1024 Megabytes	
1 Terabyte (TB)	1024 Gigabytes	

El motivo de que la proporción entre las distintas magnitudes sea de 1024, en lugar de 1000 que es lo habitual en el sistema decimal, se debe a que 1024 es la potencia de base 2 que más se aproxima al múltiplo 1000 (2^{10} = 1024), equivalente al prefijo kilo.

CÓDIGO ASCII

Como ya se ha indicado, el ordenador necesita tener los datos e instrucciones codificados en forma binaria, es decir, convertidos en 0 y 1; por tanto, **todos** los caracteres (letras, números y otros caracteres especiales del teclado) deben estar codificados mediante un código binario **unívoco**.

El código que representa a todos los caracteres disponibles en el teclado, denominado **código de caracteres,** representa cada carácter mediante un número binario constituido por un número de dígitos menor o igual que ocho; aunque, como ya justificaremos más adelante, conviene completar dichos números con ceros a la izquierda hasta formar **octetos** o **bytes** completos.

Existen distintos códigos de caracteres, pero el más utilizado sigue siendo el código **ASCII**[4]. En este sistema, a cada carácter le corresponde un número, que en el sistema decimal está comprendido entre 0 y 255 y, en el sistema hexadecimal, está comprendido entre el 00 y el FF. Cada carácter está representado, en el código ASCII, por un byte, es decir, por 8 bits.

Dependiendo del valor otorgado a un carácter, su representación binaria estará constituida por un número variable de ceros y unos. Para no mezclar dígitos de varios caracteres seguidos, el ordenador agrupa los de cada carácter en grupos completos de ocho bit, por lo que, si su código binario estuviera constituido por un número menor de dígitos, lo completaría añadiendo ceros a la izquierda. Por ejemplo, al teclear el carácter **C** (67_{10}) se introducirá y almacenará en su código binario 01000011 (43_{16}) o, al teclear la ***barra espaciadora*** (32_{10}), lo que se introduce en el registro del teclado es el código binario 00100000 (20_{16}).

Los 32 primeros caracteres del código ASCII están constituidos por los caracteres de control: ***Intro***, ***Delete***, etc. Los siguientes, hasta el 128, son caracteres internacionales y, por tanto, comunes para todos los países. De los restantes, algunos son caracteres especiales (flechas, símbolos matemáticos, etc.), y otros particulares de cada país, como, por ejemplo, nuestra característica **ñ**.

Para obtener más información visita las siguientes web:

http://www.ar.inter.net/codeas.htm

http://www.abcdatos.com/utiles/ascii.html

[4] *American Standard Code for Information Interchange*

LAS INSTRUCCIONES DE LOS PROGRAMAS

También las instrucciones de los programas, que son órdenes que indican al procesador qué debe hacer con los datos, debe estar codificadas con UNOS y CEROS. Cada procesador tiene un juego de instrucciones propio, que se ajusta a un código.

Por ejemplo, el procesador PIC 16C84, que es un pequeño procesador RISC, tiene un juego de 35 instrucciones de 14 bit de largo. A modo de ejemplo, te muestro en la tabla siguiente algunas de ellas:

ASSEMBLER	DESCRIPCIÓN	CÓDIGO
ADDWG g,d	Suma el contenido de W a G y lo envía al destino d	00 0111 dggg gggg
CLRG g	Pone a cero el registro G	00 0001 1ggg gggg
MOVWG g	Copia el contenido del registro W en G	00 0000 1ggg gggg
ADDLW K	Suma al registro W la constante K	11 111x kkkk kkkk

Si quieres consultar el resto de las instrucciones de este pequeño procesador, lo puedes hacer en la web del fabricante Microchip WebSite

UN RIO DE DATOS

Un verdadero torrente de datos circula por los canales o buses del ordenador. El procesador los recibe y debe saber reconocer si un determinado código, por ejemplo 100110101001101011011010 1, forma parte de una instrucción de un programa, constituye parte de un texto, es una cifra numérica que se necesita para un cálculo o se trata del color de un pixel de una foto.

Para que no haya confusiones, es imprescindible establecer unas normas muy estrictas o **estándares** en el modo en que se componen los códigos binarios. Todas las letras deben tener la misma longitud, todos los píxeles de una foto utilizarán la misma longitud de código para fijar el color, etc.

También va a ser necesario etiquetar los bloques de datos con códigos que le permitan saber al ordenador dónde empieza un fichero, con un código **BOF** (Beginning of File) y dónde termina, con un código **EOF** (End of File).

Y, para terminar, será necesario extremar las medidas de orden en el modo en que se almacenan los datos, utilizando códigos que etiqueten el contenido de los archivos, como las conocidas extensiones de archivo: **sxw** (documento de texto xml), **jpg** (archivo gráfico comprimido) o **html** (documento de hipertexto para la web) por ejemplo.

Para saber más, conviene que busques en Internet una relación completa de las extensiones de archivo más utilizadas y te la estudies.

EJERCICIOS

1. La codificación binaria es una de las muchas posibles. ¿Conoces otros sistemas de codificación?
2. ¿El sistema Braille es un sistema de codificación? Averigua el código Braille y su historia.
3. Averigua el código Morse y conviértelo a un código binario.
4. Calcula el código binario de cada uno de los caracteres que constituyen tu nombre. Ten en cuenta que tendrás que consultar, en una tabla ASCII, el valor decimal de cada uno de ellos.
5. Representa tu nombre completo en código hexadecimal. Para ello tendrás que unir, de forma ordenada, los octetos de los caracteres que lo forman.
6. Calcula cuántos bit se necesitan para reunir 1 GB.
7. ¿Qué dificultades plantea, en términos de cantidad de códigos necesarios, las variedad de reglas ortográficas de las distintas lenguas del planeta? ¿Es capaz el código ASCII de dar solución a toda esa variedad?