Representación de la información

1. Informática e información

La informática es la ciencia tecnológica que estudia el tratamiento automático y racional de la información, con el fin de obtener de ella la máxima utilidad. La informática usa las computadoras u ordenadores para el tratamiento y el proceso de la información. En primer lugar, y antes de saber cómo se representa la información, lo que vamos a hacer es entender bien el concepto de información.

Coloquialmente, el término información es sinónimo de conocimiento, de noticia, de datos, etc. Podríamos pensar que la información existe cuando hay comunicación de datos, pero eso no es así, en un proceso de comunicación, la información que se adquiere depende mucho del receptor. Así, por ejemplo, si mil personas escuchan una noticia en la radio, donde se transmite una gran cantidad de datos sobre lo ocurrido en el día de ayer, no todos los oyentes recibirán el mismo nivel de información. Sólo aumentará la información de aquellos oyentes que no conocen ya la noticia, mientras que para los demás la información recibida será nula. Así mismo, tampoco recibirán la información aquellos oyentes que no entienden el idioma en el que se está retransmitiendo.

Un **sistema de comunicación** está formado por los siguientes elementos básicos (véase la Figura 1.1).

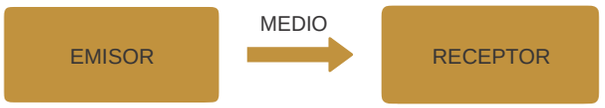

Fig. 1.1. *Elementos de un sistema de comunicación.*

- **Emisor, fuente o transmisor:** es el que genera o emite la información.
- **Receptor:** es el que recibe la información.
- **Medio o canal:** vía de transmisión de la información.

El emisor y el receptor pueden intercambiar sus papeles o incluso realizar ambos papeles de forma simultánea.

Podemos entender la transmisión de información entre el ser humano y el ordenador como una comunicación en la que el emisor es una persona y el receptor el ordenador, o viceversa, y el medio o canal son los periféricos de entrada y salida del ordenador, que son los dispositivos que se conectan al ordenador y que van a permitir introducir datos para que el ordenador los procese y transforme en forma de información.

Así pues, la relación establecida entre datos e información a través de un proceso de datos se denomina *sistema de información* (véase Figura 1.2).

Fig. 1.2. *Elementos de un sistema de información.*

Con todo esto, podemos definir **información** como la representación de hechos, objetos, valores, ideas, etc., que permiten la comunicación entre emisor y receptor, y la adquisición del conocimiento de las cosas.

Así, la **transmisión de información** entre el ser humano y la computadora puede hacerse de muchas formas:

- Mediante *caracteres alfanuméricos* (letras {a, b, ..., z} y números {0, 1, ..., 9}). Por ejemplo, los introducidos al ordenador mediante un teclado.
- Mediante *sonidos:* como los introducidos al ordenador a través de un micrófono, o que salen del ordenador por los altavoces.
- Mediante *vídeos:* como las imágenes obtenidas a través de una cámara de vídeo.
- Mediante *gráficos e imágenes:* por ejemplo, una imagen introducida por un escáner, o fotografías descargadas de una cámara de fotos digital.
- En general, cualquier tipo de dato enviado por un periférico del ordenador capaz de tomar datos de cualquier tipo y enviarlo al ordenador, o a la inversa.

En cada caso el canal es diferente, y para proceder a la comunicación de los datos es necesario cambiar la forma en que estos se representan. Podría haber hasta tres formas de representación: la del emisor, la del canal y la del receptor.

Por lo tanto, los datos deben ser traducidos o ***codificados.*** La traducción o codificación es necesaria cuando los códigos utilizados por el emisor, el canal y el receptor son diferentes.

Importante

En conclusión, para que exista información es necesario que el que envía los datos y el que lo reciba se entiendan, es decir, que utilicen el mismo código; de lo contrario, necesitarán un traductor de un código al otro.

A. Simbología y codificación

A lo largo de la historia del hombre se ha ido reconociendo que la actividad simbólica es uno de los rasgos más característicos de la actividad humana. El hombre es un creador de símbolos y a la vez un usuario de ellos, vive en un mundo simbólico de lenguajes, pensamientos, religiones, dinero, arte.

Antiguamente, en la época egipcia se empleaban símbolos para la representación de palabras (véase Figura 1.3).

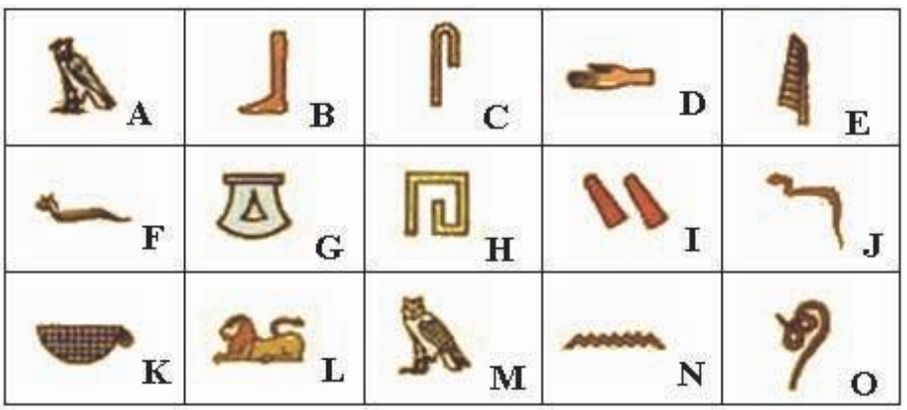

Fig. 1.3. *Símbolos egipcios para la representación de palabras.*

Dado un conjunto idóneo de símbolos, por ejemplo un vocabulario, y establecidas las reglas propias del juego, como puede ser una gramática, los símbolos pueden ser manejados como sustitutos de las cosas que representan. Esta asociación es una forma de **codificación.**

Así pues, podemos definir **codificar** como transformar unos datos a una representación predefinida y preestablecida. El abecedario es un sistema de codificación que se desarrolló para ser usado en un medio tipo plano como el papel y para poder transmitir la información a otras personas, quienes la descodifican y la convierten en pensamientos e ideas.

Signo	Código	Signo	Código	Signo	Código
A	*-	N	-*	0	-----
B	-***	O	---	1	*----
C	-*-*	P	*--*	2	**---
D	-**	Q	--*-	3	***--
E	*	R	*-*	4	****-
F	**-*	S	***	5	*****
G	--*	T	-	6	-****
H	****	U	**-	7	--***
I	**	V	***-	8	---**
J	*---	W	*--	9	----*
K	-*-	X	-**-	.	*-*-*-
L	*-**	Y	-*--	,	--**--
M	--	Z	--**	?	**--**

- : raya (señal larga) * : punto (señal corta)

Fig. 1.4. *Alfabeto o código Morse.*

Otro ejemplo de codificación es el alfabeto Morse para el telégrafo (véase Figura 1.4). Por medio de este alfabeto se transforman los datos en puntos y rayas, que son transmitidos, recibidos y descodificados hasta obtener el dato original. En este caso, el medio que sostiene los datos es una serie de impulsos eléctricos en un alambre. Aquí la codificación consiste en establecer una ley de correspondencia entre las informaciones por representar y las posibles combinaciones de puntos y rayas, de manera que a cada información le corresponda una sola configuración. Observa el alfabeto Morse en la Figura 1.4.

Llamaremos **código** a esa ley de correspondencia, es decir, al conjunto de condiciones y convenios que permiten transformar la información de una representación concreta a otra.

De este modo, un código está compuesto de:

- Un conjunto de reglas y convenios de transformación del alfabeto fuente.
- Un nuevo alfabeto que sustituirá al original.

La representación interna de la información en los ordenadores ha de darse en forma de impulsos eléctricos; esto se efectúa empleando señales biestables con dos posibles estados, activado-desactivado, encendido-apagado, abierto-cerrado, tensión-no tensión; es decir, hay impulso o no lo hay. Por eso, tendremos que codificar la información utilizando un código con dos únicos símbolos que representen los dos estados, utilizaremos el 1 para indicar que hay impulso y el 0 para indicar que no lo hay; todo el lenguaje se transcribirá a combinaciones de ceros y unos para que el ordenador lo pueda interpretar.

Este código es el **código binario,** que está basado en el sistema de numeración binario, cuyos símbolos son el 0 y el 1.

1.2. Sistemas de numeración

Se define **sistema de numeración** como el conjunto de símbolos utilizados para la representación de cantidades, así como las reglas que rigen dicha representación.

Un sistema de numeración se distingue por su **base**, que es el número de símbolos que utiliza, y se caracteriza por ser el coeficiente que determina cuál es el valor de cada símbolo dependiendo de su posición.

El sistema de numeración que utilizamos normalmente es el **sistema decimal,** de base 10. El sistema decimal utiliza diez dígitos o símbolos:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Dependiendo de la posición que ocupe un dígito dentro de una cifra, representará las unidades, decenas, centenas, millares, etc. Por esto, se dice que los sistemas de numeración son **posicionales.**

Por ejemplo, en este sistema el valor del número 6 839 se puede expresar como sumas de potencias de la **base 10:**

$$(6 \cdot 10^3) + (8 \cdot 10^2) + (3 \cdot 10^1) + (9 \cdot 10^0) = 6\,839$$

Que, de hecho, es como expresamos oralmente esta cifra:

«seis mil/ochocientos/treinta/y nueve»

Podemos definir también un sistema de numeración como un conjunto de dígitos y reglas que permiten representar datos numéricos. La **principal regla** es que un *mismo dígito tiene distinto valor según la posición que ocupe.*

Ejemplo

Si tenemos el número 555, el dígito 5 tiene distinto valor dependiendo de la posición que ocupa. Cada posición tiene un peso asociado, siendo en este caso 5 las unidades, 50 las decenas y 500 las centenas. El dígito más a la derecha tendrá peso 0, el siguiente 1, el siguiente 2, y así sucesivamente.

Podremos representar este número como las sumas de las potencias de la base 10 elevada al peso:

$(5 \cdot 10^2) + (5 \cdot 10^1) + (5 \cdot 10^0)$

A. Teorema fundamental de la numeración

Este teorema relaciona una cantidad expresada en cualquier sistema de numeración con la misma cantidad expresada en ›r decimal de una cantidad expresada en otro sist base. Viene dado por la fórmula:

$$N_i = \sum_{i=-d}^{n} (\text{dígito})_i \cdot (\text{base})^i$$

$$N_i = \sum_{i=-d}^{n} (\text{dígito})_i \cdot (\text{base})^i$$

Donde:

- i = posición respecto a la coma. Para los dígitos de la derecha; la i es negativa empezando en –1; para los de la izquierda es positiva empezando en 0.
- d = número de dígitos a la derecha de la coma.
- n = número de dígitos a la izquierda de la coma –1.
- dígito = cada uno de los que componen el número.
- base = base del sistema de numeración.

El número en decimal será el sumatorio de multiplicar cada dígito por la base elevada a su posición. i indica la posición del dígito respecto a la coma; si el número tiene comas, i se iniciará con valor negativo.

Caso práctico

En este caso práctico, vamos a ver cómo se expone el sumatorio del 6 578:

1. Calculamos los valores de la fórmula:
   - $d = 0$, no hay coma
   - $i = -d = 0$
   - $n = 3$
2. Calculamos los pesos asociados a los dígitos según la posición. El peso 0 lo tiene el dígito de la derecha, y el peso n el de la izquierda (véase Tabla 1.1).
3. Sumamos según la fórmula:

$(6 \cdot 10^3) + (5 \cdot 10^2) + (7 \cdot 10^1) + (8 \cdot 10^0)$ 6 000 + 500 + 70 + 8 = 6 578

Pesos	3	2	1	0
	10^3	10^2	10^1	10^0
Dígitos	6	5	7	8

Tabla 1.1. *Pesos asociados a la cantidad 6 578.*

B. El sistema binario

El sistema de numeración binario utiliza solo **dos dígitos** (0 y 1) para representar cantidades, por lo que su **base es 2.** Cada dígito de un número representado por este sistema se denomina **bit (bi**nary digi**t).**

Los bits tienen distinto valor dependiendo de la posición que ocupan; por eso este sistema también es posicional. Estos valores vienen determinados por una potencia de base 2 que la vamos a llamar *peso.* Así, por ejemplo, el número binario 1 011,01 expresado en decimal quedaría así:

$$(1 \cdot 2^3) + (0 \cdot 2^2) + (1 \cdot 2^1) + (1 \cdot 2^0) \rightarrow 11$$

En la Tabla 1.2 se muestran los pesos en potencia de 2 asociados según la posición del dígito. Para convertir a decimal, basta con colocar los dígitos en las columnas correspondientes y sumar los pesos donde hay un 1, hasta obtener la cantidad.

Pesos asociados				Número decimal
2^3	2^2	2^1	2^0	
8	4	2	1	
	1	1	0	6
1	0	1	1	11
1	1	0	1	13
		1	1	3
1	0	0	1	9

Tabla 1.2. *Conversión binario-decimal sumando los pesos donde hay un 1.*

Conversión de un número decimal a binario

Para representar un número en sistema binario solo podemos utilizar los dígitos 0 y 1, como hemos visto anteriormente. La forma más simple de convertir a binario es dividir sucesivamente el número decimal y los cocientes que se van obteniendo por 2 hasta que el cociente sea menor de 2. La unión del último cociente y todos los restos obtenidos escritos en orden inverso será el número expresado en binario.

Por tanto, si queremos representar el número decimal 25 en binario, realizaremos divisiones sucesivas por 2 hasta obtener un cociente menor de 2. El número resultante será **el último cociente** y tras él **los restos** obtenidos en cada una de las divisiones, empezando por el último. En la Figura 1.5 se muestra el resultado de las divisiones y el último cociente y el orden en el que deben colocarse.

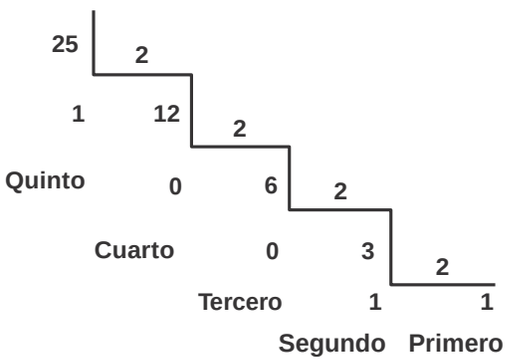

Fig. 1.5. *Conversión del número 25 a binario.*

De esta manera, **el número decimal 25** será el **11 001 en el sistema binario.**

Claves y consejos

La **cantidad de dígitos** de un número en binario dependerá del valor de dicho número en el sistema decimal. Hemos visto que para representar el número 25 necesitamos cinco **dígitos binarios.** Para representar cualquier número decimal nos guiaremos de la siguiente tabla:

Número decimal	dígitos en binario
Menor que 2	1
Menor que 4	2
Menor que 8	3
Menor que 16	4
Menor que 32	5
Menor que 64	6
Menor que 2^n	n

Tabla 1.3. *Cantidad de dígitos para representar un número decimal.*

Caso práctico 1

Expresar un número decimal en sistema binario.

Vamos a pasar a binario el número decimal 54. Para ello:

1. Calculamos el número de dígitos **N** necesarios para representar 54. El número 54 es mayor que $2^5 = 32$, pero es menor que $2^6 = 64$; entonces, con **seis dígitos binarios** podremos representar el **número decimal 54.**

2. Podremos realizar una tabla tal como la 1.4, con los seis dígitos, y luego sumar los pesos donde hay un 1, como muestra la Tabla 1.6.

Pesos asociados							
2	2	2	2	2	2	2	2
128	6	3	1	8	4	2	1
		1	1	0	1	1	0
		32 +	16	0 +	4 +	2	0 → 54

Tabla 1.4. *Conversión a binario de 54 mediante suma de pesos.*

3. O bien realizamos divisiones sucesivas del número 54 por 2 hasta llegar a un cociente menor de 2 (véase Figura 1.6).

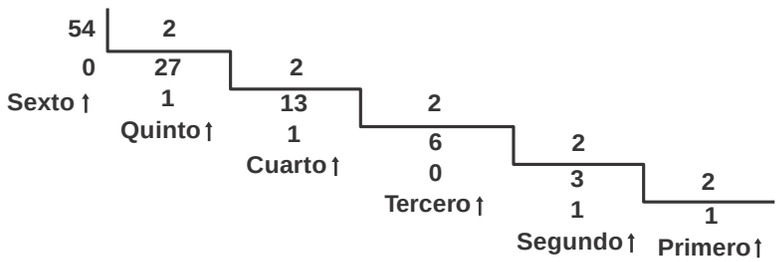

Fig. 1.6. *Conversión a binario de 54 mediante divisiones sucesivas.*

Por tanto, el número decimal 54 se representa en código binario como **110110.** Lo escribimos así:

$$54_{(10} \rightarrow 110110_{(2}$$

C. El sistema octal

Los primeros sistemas informáticos utilizaban solo el sistema binario para interpretar y transformar los datos, con lo que las labores de programación eran bastante tediosas; se recurrió entonces al uso de sistemas intermedios que permitían una fácil traducción hacia y desde el sistema binario. Estos sistemas son el *octal* y el *hexadecimal*.

El **sistema octal** tiene como base de numeración 8, es decir, utiliza ocho símbolos para representar las cantidades. Estos símbolos son 0, 1, 2, 3, 4, 5, 6, 7. Este sistema también es posicional; o sea, que un mismo dígito tiene distinto valor según la posición que ocupe. Para convertir de decimal a octal, y viceversa, procederemos como en el sistema binario:

- **Conversión de un número decimal a octal.** Lo más sencillo son las divisiones sucesivas. En la Figura 1.7 se convierte a octal el número 925.

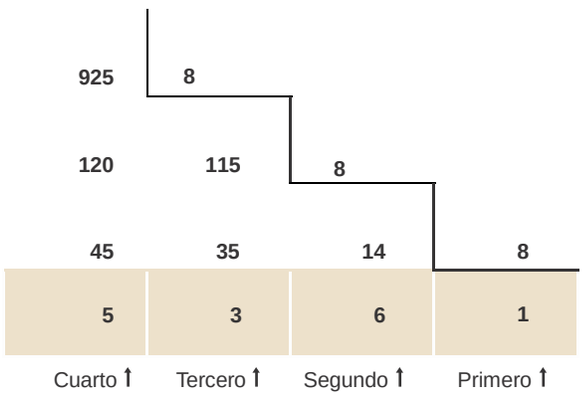

Fig. 1.7. *Conversión a octal de 925 mediante divisiones sucesivas.*

- Para **convertir un número octal a decimal,** emplearemos el teorema fundamental de la numeración. Nos podremos guiar por los pesos asociados a cada dígito dependiendo de su posición. En la Tabla 1.5 se muestra la cantidad 1 635 en octal. Para pasar a decimal, multiplicamos el dígito por la base elevada a su posición:

Pesos asociados en el sistema octal			
8^3	8^2	8^1	8^0
512	64	8	1
1	**6**	**3**	**5**

Tabla 1.5. *Pesos asociados en el sistema octal.*

$$(1 \cdot \mathbf{8^3}) + (6 \cdot \mathbf{8^2}) + (3 \cdot \mathbf{8^1}) + (5 \cdot \mathbf{8^0}) \rightarrow$$

$$\rightarrow (1 \cdot \mathbf{512}) + (6 \cdot \mathbf{64}) + (3 \cdot \mathbf{8}) + (5 \cdot \mathbf{1}) \rightarrow \mathbf{925}$$

$$\mathbf{1635_{(8} \rightarrow 925_{(10}}$$

D. El sistema hexadecimal

El **sistema hexadecimal** tiene como base de numeración 16, es decir, utiliza dieciséis símbolos para representar las cantidades. Estos símbolos son 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Este sistema también es posicional. A los símbolos A, B, C, D, E y F se les asignan los valores que se muestran en la Tabla 1.6.

Símbolo	Valor asignado
A	10
B	11
C	12
D	13
E	14
F	15

Tabla 1.6. *Sistema hexadecimal: valores asignados a los símbolos A, B, C, D, E y F.*

Para convertir de hexadecimal a decimal, y viceversa, procederemos como en los casos anteriores.

- **Conversión de un número decimal a hexadecimal.** Se realizan divisiones sucesivas, y para los restos entre 10 y 15 utilizamos las letras correspondientes, como se muestra en la Tabla 1.6. En la Figura 1.8 se convierte el número 41 565 a hexadecimal.

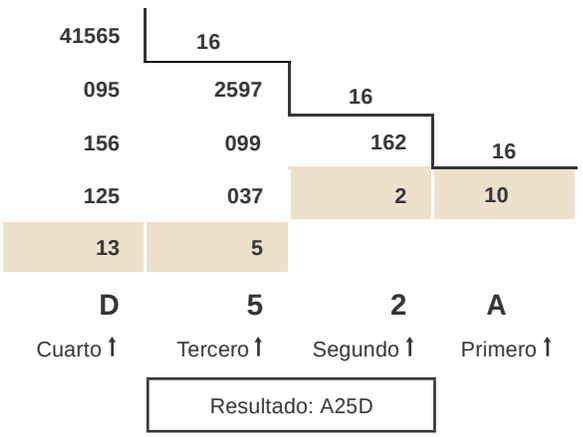

Fig. 1.8. *Conversión a hexadecimal de 41 565.*

- Para **convertir un número hexadecimal a decimal,** utilizaremos el teorema fundamental de la numeración. En la Tabla 1.7 se muestran los pesos asociados por cada posición. Para pasar la cantidad $A1D_{(16}$ a decimal, multiplicamos el dígito por la base elevada a su posición:

$$(A \cdot 256) + (1 \cdot 16) + (D \cdot 1) = (10 \cdot 256) + 16 + 13 = 2\,560 + 16 + 13 = 2\,589$$

$$\mathbf{A1D_{(16} = 2589_{(10}}$$

Pesos asociados en el sistema hexadecimal			
16^3	16^2	16^1	16^0
4096	256	16	1
	A	1	D

Tabla 1.7. *Pesos para la conversión hexadecimal-decimal.*

E. Conversiones entre sistemas

De la misma manera que convertimos del sistema decimal al binario, octal y hexadecimal, y viceversa. También podemos convertir del binario al octal y hexadecimal y del hexadecimal al octal, etc. (véase Tabla 1.8).

DEC	BIN	OCT	HEX
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**
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14
21	10101	25	15
22	10110	26	16
23	10111	27	17
24	11000	30	18
25	11001	31	19
26	11010	32	1A
27	11011	33	1B
28	11100	34	1C
29	11101	35	1D
30	11110	36	1E

Tabla 1.8. *Equivalencias entre sistemas decimal, binario, octal y hexadecimal.*

Conversión hexadecimal-binario

Se sustituye cada dígito hexadecimal (0, 1, 2, …, D, E, F) por su representación binaria utilizando cuatro dígitos; así, el 0 se representa por 0000, el 1 por 0001, el 2 por 0010, etc. Se utilizan cuatro dígitos porque el valor más alto de este código, el 15, que se representa con la F, necesita cuatro dígitos: 1111. Véase Caso práctico 14.

Caso práctico 2

Conversión de hexadecimal a binario.

Pasar a binario $73B_{(16}$:

7 - 3 - B
↓ ↓ ↓
0111 0011 1011

$73B_{(16}$ → $11100111011_{(2}$

Conversión binario-hexadecimal

Se agrupan los dígitos binarios de cuatro en cuatro a partir del punto decimal hacia la izquierda y hacia la derecha, y se sustituye cada grupo de cuatro por su valor correspondiente en hexadecimal. Véase Caso práctico 15.

Caso práctico 3

Pasar a hexadecimal $101011011_{(2}$:

0001 - 0101 - 1011
↓ ↓ ↓
1 5 B

$101011011_{(2}$ → $15B_{(16}$

Conversión octal-binario

Procedemos como en la conversión hexadecimal-binario; se sustituye cada dígito octal por su representación binaria utilizando tres dígitos binarios. Se utilizan tres porque el valor más alto, el 7, necesita tres dígitos binarios: 111. Véase Caso práctico 16.

Caso práctico 4

Pasar a binario $527_{(8}$:

5 - 2 - 7
↓ ↓ ↓
101 010 111

$\mathbf{527_{(8}} \rightarrow 101010111_{(2}$

Pasar a binario $712_{(8}$:

7 - 1 - 2
↓ ↓ ↓
111 001 010

$\mathbf{712_{(8}} \rightarrow \mathbf{111001010_{(2}}$

Conversión binario-octal

Se agrupan los dígitos de tres en tres a partir del punto decimal hacia la izquierda y hacia la derecha, sustituyendo cada grupo de tres por su equivalente en octal. Véase Caso práctico 17.

Caso práctico 5

Pasar a octal $10101100_{(2}$:

010 - 101 - 100
↓ ↓ ↓
2 5 4

$\mathbf{10101100_{(2}} \rightarrow \mathbf{254_{(8}}$

Pasar a octal $1110110_{(2}$:

001 - 110 - 110
↓ ↓ ↓
1 6 6

$\mathbf{1110110_{(2}} \rightarrow \mathbf{166_{(8}}$

Conversión hexadecimal-octal

En esta conversión se realiza un paso intermedio; primero se pasa de hexadecimal a binario y luego de binario a octal. Véase Caso práctico 18.

Caso práctico 6

Pasar $1AB0C_{(16}$ a octal.

– Convertir a binario $1AB0C_{(16}$:

1 A B 0 C
↓ ↓ ↓ ↓ ↓
0001 1010 1011 0000 1100

– Convertir a octal $11010101100001100_{(2}$

011 010 101 100 001 100
↓ ↓ ↓ ↓ ↓ ↓
3 2 5 4 1 4

$\mathbf{1AB0C_{(16}} \rightarrow \mathbf{325414_{(8}}$

Conversión octal-hexadecimal

Se realiza como la anterior, pero en este caso, primero se pasa de octal a binario y luego de binario a hexadecimal. Véase Caso práctico 19.

Caso práctico 7

Pasar $3710_{(8}$ a hexadecimal.

1. Convertir a binario $3710_{(8}$:

3	7	1	0
↓	↓	↓	↓
011	111	001	000

2. Convertir a hexadecimal $11111001000_{(2}$:

111	1100	1000
↓	↓	↓
7	C	8

$3710_{(8}$ → $7C8_{(16}$

Operaciones Lógicas

Se basan en el Álgebra de Boole. Las operaciones lógicas básicas (.NOT, .AND, .OR, .XOR, .NAND, .NOR)

Tabla de operadores lógicas:

.NOT			.AND			.OR			.XOR			.NAND			.NOR		
A	=		A	B	=	A	B	=	A	B	=	A	B	=	A	B	=
0	**1**		0	0	**0**	0	0	**0**	0	0	**0**	0	0	1	0	0	**1**
1	**0**		0	1	**0**	0	1	**1**	0	1	**1**	0	1	1	0	1	**0**
			1	0	**0**	1	0	**1**	1	0	**1**	1	0	1	1	0	**0**
			1	1	**1**	1	1	**1**	1	1	**0**	1	1	0	1	1	**0**

.NOT: El resultado será la negación de los valores.
.AND: El resultado será 1 solo cuando ambos sean 1.
.OR: El resultado será 1 cuando uno de ambos valores sea 1.
.XOR: El resultado será 0 cuando ambos valores sean iguales y 1 cuando sean diferentes.
.NAND: Es la negación de AND.

.NOR: Es la negación de OR.

Ejemplos:

	11101100	01101100
	00011101	00101101
AND	00001100	00101100
OR	11111101	01101101
XOR	11110001	01000001
NAND	11110011	11010011
NOR	00000010	10010010

A	B	A XOR B	A AND B	NOT A	NOT A AND B
0	0	**0**	**0**	**1**	**0**
1	0	**1**	**0**	**0**	**0**
0	1	**1**	**0**	**1**	**1**
1	1	**0**	**1**	**0**	**0**

1.3. Representación interna de la información

Importante

Ya hemos visto que en informática se utiliza el sistema binario, solo se manejan las cifras cero y uno (0 y 1), los ordenadores trabajan internamente con dos niveles de voltaje: «apagado» (0) y «encendido» (1), por lo que su sistema de numeración natural es el sistema binario.

El **bit** es la unidad mínima de almacenamiento empleada en informática, en cualquier dispositivo digital o en la teoría de la información; con él podemos representar dos valores cualesquiera, como verdadero o falso, abierto o cerrado, blanco o negro, norte o sur, rojo o azul… Basta con asignar uno de esos valores al estado de «apagado» (0) y el otro al estado de «encendido» (1).

Cuando se almacena la información no se trabaja a nivel de bit, sino que se trabaja a nivel de carácter (letra, número o signo de puntuación), que ocupa lo que se denomina un **byte,** que a su vez está compuesto de 8 **bits**. El ordenador trabaja con agrupaciones de bits fáciles de manipular y suelen ser múltiplos de 2, la base del sistema binario. Los tamaños más comunes son:

- **Octeto, carácter** o **byte:** es la agrupación de 8 bits, el tamaño típico de información; con él se puede codificar el alfabeto completo (ASCII estándar).
- **Palabra:** tamaño de información manejada en paralelo por los componentes del sistema, como la memoria, los registros o los buses. Son comunes las palabras de 8, 32, 64, 128 y 256 bits: 1 byte, 4, 8, 16, 32 bytes. A mayor tamaño de palabra, mayor es la precisión y la potencia de cálculo del ordenador.

Así, cuando decimos que un archivo de texto ocupa 5 000 bytes, queremos decir que contiene el equivalente a 5 000 letras o caracteres (entre dos y tres páginas de texto sin formato).

Lo normal es utilizar los múltiplos del byte: el kilobyte (kb), el megabyte (Mb), el gigabyte (Gb), etc.

En informática se utilizan las potencias de 2 (2^3, 2^{10}, 2^{20}…) para representar las medidas de la información; sin embargo se ha extendido el uso de las potencias de 10 (uso decimal), debido a que se ha impuesto el uso del *Sistema Internacional de Medidas* (SI), o sistema métrico. Así pues, el primer término de medida que se utilizó fue el **kilobyte** (kb), y se eligió este porque 2^{10} es aproximadamente 1 000, que se asocia con el kilo (1 000 gramos); en realidad debería ser 1 024 bytes, ya que 2^{10} son 1 024.

La Tabla 1.9 muestra las unidades de medida de información más utilizadas, tanto en su uso decimal como en su uso binario:

Nombre (símbolo)	Sistema Internacional de Unidades (SI) Estándar (uso decimal)	Prefijo binario (uso binario)	Nombre (símbolo)
Kilobyte (kb)	$1000^1 = 10^3$ bytes	$1024^1 = 2^{10}$ bytes	Kibibyte (kib)
Megabyte (Mb)	$1000^2 = 10^6$ bytes	$1024^2 = 2^{20}$ bytes	Mebibyte (Mib)
Gigabyte (Gb)	$1000^3 = 10^9$ bytes	$1024^3 = 2^{30}$ bytes	Gibibyte (Gib)
Terabyte (Tb)	$1000^4 = 10^{12}$ bytes	$1024^4 = 2^{40}$ bytes	Tebibyte(Tib)
Petabyte (Pb)	$1000^5 = 10^{15}$ bytes	$1024^5 = 2^{50}$ bytes	Pebibyte (Pib)
Exabyte (Eb)	$1000^6 = 10^{18}$ bytes	$1024^6 = 2^{60}$ bytes	Exbibyte (Eib)
Zettabyte (Zb)	$1000^7 = 10^{21}$ bytes	$1024^7 = 2^{70}$ bytes	Zebibyte (Zib)
Yottabyte (Yb)	$1000^8 = 10^{24}$ bytes	$1024^8 = 2^{80}$ bytes	Yobibyte (Yib)

Tabla 1.9. *Unidades de medida de información en decimal y en binario.*

El **megabyte** (Mb). Equivale a 10^6 (1 000 000 bytes) o 2^{20} (1 048 576 bytes), según el contexto. Es el conjunto de 1 024 kilobytes, 2^{20} bytes $\rightarrow 2^{10} \cdot 2^{10} = 1\,024 \cdot 1\,024 \rightarrow$ $\rightarrow 1\,048\,576$; también podemos decir un millón de bytes 10^6.

Un **gigabyte** (Gb) equivale a 2^{30} bytes o 10^9 bytes, según el uso. Es la unidad que más se usa actualmente para especificar la capacidad de la memoria RAM, de las memorias de tarjetas gráficas, de los CD-ROM, o el tamaño de los programas, de los archivos grandes. La capacidad de almacenamiento se mide habitualmente en gigabytes, es decir, en miles de megabytes. Un Gb es el conjunto de 1024 megabytes, 2^{30} bytes, o lo que es lo mismo, $2^{10} \cdot 2^{10} \cdot 2^{10} = 1\,024 \cdot 1\,024 \cdot 1\,024 \rightarrow 1\,073\,741\,824$; mil millones de bytes 10^9.

Medida de la información relacionada con la velocidad de transmisión

En informática, es muy frecuente que, además de la capacidad que tienen los dispositivos para almacenar información, sea necesario medir la velocidad a la que puede enviarse la información de un dispositivo a otro (también conocida como bit rate). En este contexto, se suele hablar de la cantidad de bits que pueden transmitirse en cada segundo (bits por segundo, bps o b/s).
Como múltiplo, podemos utilizar bytes por segundo (B/s), que se corresponderá a la velocidad en bps dividida entre ocho.
Por encima de estos valores, podemos encontrar los siguientes múltiplos:

Unidad	Abreviatura	Nº de bps
kilobit por segundo	kbps, kbit/s ó kb/s	10^3
kilobyte por segundo	kBps ó kB/s	8×10^3
megabit por segundo	Mbps, Mbit/s ó Mb/s	10^6
megabyte por segundo	MBps ó MB/s	8×10^6
gigabit por segundo	Gbps, Gbit/s ó Gb/s	10^9
gigabyte por segundo	GBps ó GB/s	8×10^9
terabit por segundo	Tbps, Tbit/s ó Tb/s	10^{12}
terabyte por segundo	TBps ó TB/s	8×10^{12}

A modo de ejemplo del uso de las medidas de información relacionadas tanto con el tamaño de los dispositivos, como con la velocidad de transmisión, veamos un ejemplo típico que anuncia las características de un dispositivo de almacenamiento externo:

Disco duro de 128Gb - OCZ Vector SSD

Características

- Capacidad de 128GB.
- Velocidad de lectura de 550MB/s.
- Velocidad de escritura de 400MB/s.
- Conexión SATA 3.0 (6Gb/s).

A. Representación de datos alfabéticos y alfanuméricos

Ya hemos visto cómo se almacenan las cantidades numéricas dentro del ordenador; ahora nos toca ver cómo se almacena el resto de caracteres que forman el alfabeto.

Los códigos de E/S permitirán traducir la información o los datos que nosotros podemos entender a una representación que la máquina puede interpretar y procesar. Los datos llegan y salen del ordenador a través de los periféricos de entrada y de salida, respectivamente. Cada fabricante de componentes de E/S podría asignar una combinación diferente al mismo símbolo de origen (por ejemplo, las letras del alfabeto); sin embargo, esto no sería nada positivo en un mercado abierto como el informático. Por eso se tiende a la estandarización de códigos, que ha llevado a la universalización de unos pocos códigos de E/S, como el BCD, EBCDIC, ASCII y Unicode. La mayoría de estos códigos representan cada carácter por medio de un byte (8 bits). Sin duda, el más importante de todos estos es el ASCII.

ASCII

El **Código Estadounidense Estándar para el Intercambio de Información**, o ASCII *(American Standard Code for Information Interchange)*, es la recomendación X3.4-1977 del Instituto Estadounidense de Normas Nacionales (ANSI). Utiliza grupos de 7 bits por carácter, permitiendo $2^7 \rightarrow 128$ caracteres diferentes, lo que es suficiente para el alfabeto en letras mayúsculas y minúsculas y los símbolos de una máquina de escribir corriente, además de algunas combinaciones reservadas para su uso interno. El código *ASCII extendido* usa 8 bits por carácter, lo que añade otros 128 caracteres posibles. Este juego de códigos más amplio permite que se agreguen los símbolos de lenguajes extranjeros y varios símbolos gráficos.

¿Sabías que...?

ASCII también se conoce como la ISO 8859-1 y es el utilizado por los sistemas operativos MS-DOS, Windows y UNIX. En las Tablas 1.10 y 1.11 pueden verse el código ASCII y el ASCII extendido, sin los caracteres de control, que son los primeros 32 caracteres, del 0 al 31.

Código decimal	Carácter ASCII	Código decimal	Carácter ASCII	Código decimal	Carácter ASCII	Código decimal	Carácter ASCII	Código decimal	Carácter ASCII	Código decimal	Carácter ASCII
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	_
48	0	64	@	80	P	96	`	112	p		

Tabla 1.10. *Código Standard ASCII (caracteres alfanuméricos).*

Cód.	Carác.	Cód.	Carác.	Cód.	Carác.	Cód.	Carác.	Cód.	Carác.	Cód.	Carác.	Cód.	Carác.	Cód.	Carác.
128	€	144		160		176	°	192	À	208	Đ	224	à	240	ð
129		145	‘	161	¡	177	±	193	Á	209	Ñ	225	á	241	ñ
130	‚	146	’	162	¢	178	²	194	Â	210	Ò	226	â	242	ò
131	*ƒ*	147	″	163	£	179	³	195	Ã	211	Ó	227	ã	243	ó
132	„	148	″	164	¤	180	´	196	Ä	212	Ô	228	ä	244	ô
133	…	149	•	165	¥	181	µ	197	Å	213	Õ	229	å	245	õ
134	†	150	–	166	¦	182	¶	198	Æ	214	Ö	230	æ	246	ö
135	‡	151	—	167	§	183	·	199	ç	215	×	231	ç	247	÷
136	ˆ	152	˜	168	¨	184	¸	200	È	216	Ø	232	è	248	ø
137	‰	153	™	169	©	185	¹	201	É	217	Ù	233	é	249	ù
138	Š	154	š	170	ª	186	V	202	Ê	218	Ú	234	ê	250	ú
139	‹	155		171	«	187	»	203	Ë	219	Û	235	ë	251	û
140	Œ	156	œ	172	¬	188	¼	204	Ì	220	Ü	236	ì	252	ü
141		157		173		189	½	205	Í	221	Ý	237	í	253	ý
142	Ž	158	ž	174	®	190	¾	206	Î	222	Þ	238	î	254	þ
143		159	Ÿ	175	¯	191	¿	207	Ï	223	ß	239	ï	255	ÿ

Tabla 1.11. *Código Standard ASCII extendido (caracteres alfanuméricos).*

Unicode

El **Unicode Standard** es una norma de codificación universal de caracteres que se em- plea en los ordenadores bajo Windows NT y en los navegadores Internet Explorer y Netscape a partir de su versión 4. Su uso se está extendiendo. Utiliza 16 bits, lo que permite codificar todos los caracteres de cualquier lenguaje, hasta 65 536.

La versión 3 de Unicode tiene 49194 caracteres de los utilizados en los lenguajes más importantes del mundo. El objetivo de Unicode es representar cada elemento usado en la escritura de cualquier idioma del planeta. Los idiomas actuales más importantes del

mundo pueden escribirse con Unicode, incluyendo su puntuación, símbolos especiales, símbolos matemáticos y técnicos, formas geométricas, caracteres gráficos y modelos de Braille.

Unicode proporciona un número único para cada carácter, sin importar la plataforma, sin importar el programa, sin importar el idioma. Líderes de la industria tales como Apple, HP, IBM, JustSystem, Microsoft, Oracle, SAP, Sun, Sybase, Unisys y muchos otros han adoptado la norma Unicode. Unicode es un requisito para los estándares modernos tales como XML, Java, ECMAScript (JavaScript), LDAP, CORBA 3.0, WML, etc., y es la manera oficial de aplicar la norma ISO/IEC 10646. Es compatible con numerosos sistemas operativos, con todos los exploradores actuales y con muchos otros productos. La aparición de la norma Unicode y la

disponibilidad de herramientas que la respal- dan se encuentran entre las más recientes e importantes tendencias en tecnología de software.

La incorporación de Unicode en sitios web y en aplicaciones de cliente-servidor o de múltiples niveles permite disminuir ostensiblemente los costos del uso de juegos de ca- racteres heredados. Unicode permite que un producto de software o sitio web especí- fico se oriente a múltiples plataformas, idiomas y países, sin necesidad de rediseñarlo. Además, permite que los datos se trasladen a través de gran cantidad de sistemas distintos sin sufrir daños.

Básicamente, las computadoras solo trabajan con números. Almacenan letras y otros caracteres mediante la asignación de un número a cada uno. Antes de que se inven- tara Unicode, existían cientos de sistemas de codificación distintos para asignar estos números. Ninguna codificación específica podía contener caracteres suficientes; por ejemplo, la Unión Europea, por sí sola, necesita varios sistemas de codificación distintos para cubrir todos sus idiomas. Incluso para un solo idioma como el inglés no había un único sistema de codificación que se adecuara a todas las letras, signos de puntuación y símbolos técnicos de uso común.

Síntesis

Test de repaso

1. ¿Indica cuál de los siguientes números no está codificado en octal?:

a) 12345,678

b) 234,001

c) 2347,0011

d) 3221,02

2. Si el ancho de palabra es de 10 bits, ¿cuántos números podremos representar?:

a) 100

b) 1000

c) 1024

d) 10

3. ¿Cuántos dígitos binarios necesito para representar el número 43?:

a) 5

b) 6

c) 4

d) 7

4. ¿Cuántos bytes tienen tres gigabytes?:

a) Tres millones de bytes.

b) Tres mil millones de bytes.

c) Tres mil kilobytes.

d) Trescientos millones de bytes.

5. El número 36 en octal se representa en binario a:

a) 00110110

b) 11001001

c) 011110

d) 100001

6. Para representar caracteres alfabéticos y alfanuméricos, utilizaremos el código:

a) ANSI

b) Binario

c) ASCII

d) IEEE754

7. El código EBCDIC es el utilizado por:

a) Los ordenadores IBM de la serie IBM PC.

b) Los ordenadores bajo Windows NT.

c) Los equipos de la marca Compaq.

d) Los navegadores de Internet.

8. De los siguientes códigos, ¿cuál es el que utiliza la mayoría de los navegadores de Internet?:

a) EBCDIC

b) BCD

c) Unicode

d) ASCII

9. De los siguientes códigos, ¿cuál es el que utiliza la mayoría de los navegadores de Internet?:

a) EBCDIC

b) BCD

c) Unicode

d) ASCII

10. ¿Cuántos bits tienen 12 kb?:

a) $12 \cdot 1\,024 \rightarrow 12\,288$ bits.

b) $12 \cdot 1\,024 \cdot 8 \rightarrow 98\,304$ bits.

c) $12 \cdot 1\,000 \rightarrow 12\,000$ bits.

d) $12 \cdot 1\,000 \cdot 8 \rightarrow 9\,600$ bits.

11. El número decimal 34 se representa en binario como:

a) 100100

b) 100010

c) 100001

d) 100011

12. El número binario 1101 equivale al número decimal:

a) 23

b) 14

c) 15

d) 13

Actividades

I. Sistemas de numeración

1. Expresa la cantidad según el teorema fundamental de la numeración.

- 234,765
- 347,21
- 800,102

2. Representa en el sistema decimal los siguientes números en distintas bases:

- $123_{(6}$
- $4300_{(5}$
- $1101_{(2}$

3. Convierte a binario:

- $178_{(8}$
- $29_{(10}$
- $A_{(16}$

4. Convierte a hexadecimal:

- $110010_{(2}$
- $56_{(10}$
- $156_{(8}$

5. Convierte a octal:

- $9A_{(16}$
- $29_{(10}$
- $1101110_{(2}$

II. Códigos alfanuméricos utilizados por los ordenadores

6. Codifica en ASCII y EBCDIC las palabras:

- Instalación
- Mantenimiento.

III. Medidas de almacenamiento de la información en el ordenador

7. Expresa en bytes las siguientes cantidades:

- 25 Yb
- 15 Zb
- 20 Pb

IV. Operaciones lógicas

8. Siendo los valores de A=0, B=1, C=0, D=1 calcular el resultado de la función lógica siguiente:
(not ((A and B) or C)) nand D

9. ¿Qué valores toma la función lógica (A nand B) or (not C) para los siguientes valores de A,B y C
a) A=0, B= 0, C=0
b) A=1, B=1, C=1

10. Dada la siguiente función lógica: not ((A and B) nor C) ¿qué valor o valores puede tener A si el valor de B es 0, de C es 1 y el resultado de la función es 0?.