EJERCICIOS DEL TEMA 1

Introducción a los ordenadores

1) Averigua y escribe el código ASCII correspondiente, tanto en decimal como en binario, a las letras de tu nombre y apellidos. Distinguir entre mayúsculas/minúsculas, y sin acentos.

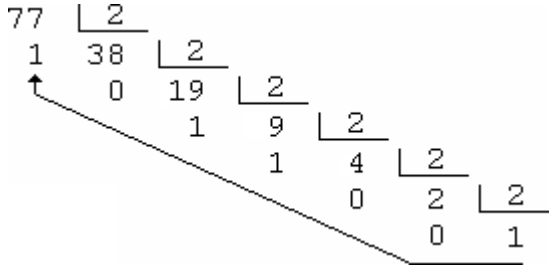
Nota: Al final de este documento podéis encontrar la tabla ASCII de los caracteres imprimibles.

LETRA	DECIMAL	BINARIO
M	77	1001101
a	97	1100001
n	110	1101110
u	117	1110101
e	101	1100101
l	108	1101100
<espacio>	32	0100000
G	71	1000111
i	105	1101001
l	108	1101100
<espacio>	32	0100000
P	80	1010000
e	101	1100101
r	114	1110010
e	101	1100101
z	122	1111010

Para cada letra del nombre extraemos su correspondencia en decimal utilizando la tabla ASCII. Una vez tenemos todos los valores decimales del nombre procedemos a convertir cada valor decimal a binario.

Como ejemplo, veamos la conversión de la primera letra del nombre (M) a su valor en binario.

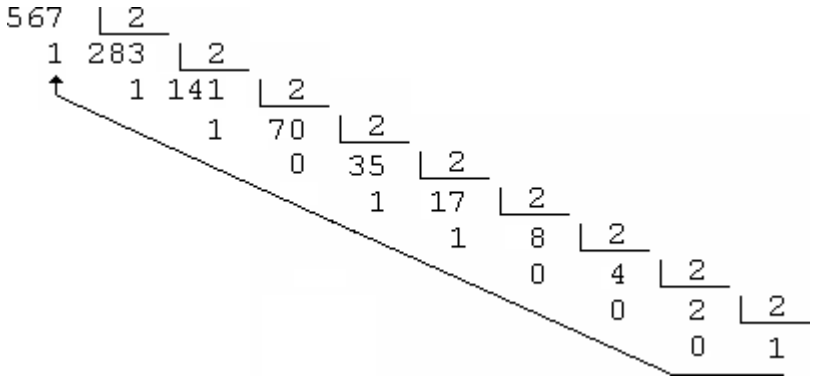
- **Conversión a binario del número 77_{10}**



El resultado, siguiendo todos los restos empezando por el cociente de más a la derecha, nos da el resultado final en binario. Es decir, $\mathbf{1001101_2}$. La conversión para el resto de letras se realizaría siguiendo este mismo proceso.

2) Realiza la conversión a binario del número decimal 567.

Siguiendo el mismo proceso que en el ejercicio anterior, tenemos que dividir sucesivamente el 567_{10} entre 2 hasta que lleguemos a un número que ya no sea divisible entre 2.



Siguiendo todos los restos obtenidos, empezando por el cociente de más a la derecha, nos da el resultado final en binario. Es decir, $\mathbf{1000110111_2}$.

3) Realiza la conversión tanto a decimal como a hexadecimal del número binario 1001110110.

- **Conversión a decimal**

En esta conversión cada cifra será multiplicada por la base del sistema de numeración (b=2) elevada a una potencia que dependerá de la posición de esa cifra en el número a convertir. Empezando por la posición de más a la derecha la primera cifra se multiplica por 2^0, la segunda por la derecha se multiplica por 2^1, y así sucesivamente.

$\mathbf{1001110110}_2 = \mathbf{1}\times2^9 + \mathbf{0}\times2^8 + \mathbf{0}\times2^7 + \mathbf{1}\times2^6 + \mathbf{1}\times2^5 + \mathbf{1}\times2^4 + \mathbf{0}\times2^3 + \mathbf{1}\times2^2 + \mathbf{1}\times2^1 + \mathbf{0}\times2^0$

$1001110110_2 = 1\times2^9 + 1\times2^6 + 1\times2^5 + 1\times2^4 + 1\times2^2 + 1\times2^1$ (obviamos los ceros)

$1001110110_2 = 512 + 64 + 32 + 16 + 4 + 2$

$1001110110_2 = \mathbf{630_{10}}$

- **Conversión a hexadecimal**

Para la conversión a hexadecimal el número a convertir se divide en grupos de 4 bits (empezando por la derecha). Si el último grupo no tiene 4 bits se le añaden tantos ceros a la izquierda como sea necesario hasta completar el grupo.

Por tanto, el número 1001110110_2 lo dividimos en 3 grupos de 4 bits cada uno de ellos; a saber: 10, 0111 y 0110. Como el último grupo no llega a los 4 bits lo rellenamos con ceros a la izquierda, quedando los 3 grupos de 4 bits como: 0010, 0111 y 0110.

BINARIO	DECIMAL	HEXADECIMAL
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F

Tabla 1: Conversión directa entre binario, decimal y hexadecimal

Después de esa división, la conversión es directa (ver Tabla 1) ya que a cada grupo de 4 bits (2^4 = 16 posibles valores = los que tiene el alfabeto hexadecimal) le corresponde un valor en el alfabeto hexadecimal.

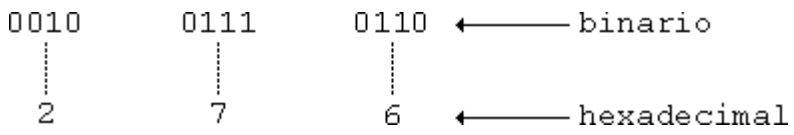
La conversión de cada grupo, si no sabemos la conversión directa entre el binario y el hexadecimal, se haría de la siguiente manera:

$\mathbf{0010}_2 = \mathbf{0}\times2^3 + \mathbf{0}\times2^2 + \mathbf{1}\times2^1 + \mathbf{0}\times2^0 = 1\times2^1 = 2_{10} = 2_{16}$

$\mathbf{0111}_2 = \mathbf{0}\times2^3 + \mathbf{1}\times2^2 + \mathbf{1}\times2^1 + \mathbf{1}\times2^0 = 1\times2^2 + 1\times2^1 + 1\times2^0 = 4 + 2 + 1 = 7_{10} = 7_{16}$

$\mathbf{0110}_2 = \mathbf{0}\times2^3 + \mathbf{1}\times2^2 + \mathbf{1}\times2^1 + \mathbf{0}\times2^0 = 1\times2^2 + 1\times2^1 = 4 + 2 = 6_{10} = 6_{16}$

Por tanto, como podemos ver en la siguiente figura, al 0010_2 le corresponde el valor hexadecimal 2, al 0111_2 el valor hexadecimal 7 y al 0110_2 el valor hexadecimal 6.



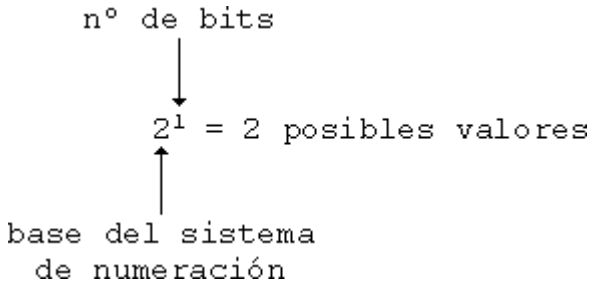
El resultado final es: 1001110110_2 = $\mathbf{276_{16}}$.

4) El sistema octal es un sistema de numeración en base 8 (b=8) cuyo alfabeto es { 0, 1, 2, 3, 4, 5, 6, 7 }. Construir una tabla con la representación de los 32 primeros números en los sistemas de numeración hexadecimal, decimal, octal y binario.

BINARIO	OCTAL	DECIMAL	HEXADECIMAL	BINARIO	OCTAL	DECIMAL	HEXADECIMAL
0000	00	00	00	10000	20	16	10
0001	01	01	01	10001	21	17	11
0010	02	02	02	10010	22	18	12
0011	03	03	03	10011	23	19	13
0100	04	04	04	10100	24	20	14
0101	05	05	05	10101	25	21	15
0110	06	06	06	10110	26	22	16
0111	07	07	07	10111	27	23	17
1000	10	08	08	11000	30	24	18
1001	11	09	09	11001	31	25	19
1010	12	10	0A	11010	32	26	1A
1011	13	11	0B	11011	33	27	1B
1100	14	12	0C	11100	34	28	1C
1101	15	13	0D	11101	35	29	1D
1110	16	14	0E	11110	36	30	1E
1111	17	15	0F	11111	37	31	1F

Tabla 2: Los 32 primeros números en varios sistemas de numeración

Como se puede comprobar en la Tabla 2, hay cierta relación en el número de bits que necesitan los diferentes sistemas de numeración vistos en clase. Con un bit, el sistema binario puede codificar hasta 2 valores; el 0 y el 1.



Con 2 bits se pueden llegar a codificar hasta $2^2 = 4$ posibles valores diferentes:

BINARIO 2 DÍGITOS	**DECIMAL**
00	0
01	1
10	2
11	3

Tabla 3: Los 4 posibles valores que se pueden codificar con 2 bits

Por tanto, un sistema de numeración cuyo alfabeto fuese { 0, 1, 2, 3 } (4 posibles valores diferentes) podría codificarse en un ordenador utilizando únicamente 2 bits. Con 3 bits se pueden codificar hasta $2^3 = 8$ posibles valores diferentes:

BINARIO 3 DÍGITOS	**OCTAL** 1 DÍGITO	**DECIMAL**
000	0	0
001	1	1
010	2	2
011	3	3
100	4	4
101	5	5
110	6	6
111	7	7

Tabla 4: Los 8 posibles valores que se pueden codificar con 3 bits

Por tanto, con 3 bits podemos codificar todos los posibles valores del sistema de numeración octal (b=8); es decir, todos los posibles valores de su alfabeto { 0, 1, 2, 3, 4, 5, 6, 7 } (8 posibles valores diferentes).

Por último, con 4 bits se pueden codificar hasta $2^4 = 16$ posibles valores diferentes; es decir, los mismos posibles valores que tiene el sistema hexadecimal (b=16). Ver la Tabla 5.

BINARIO 4 DÍGITOS	**HEXADECIMAL** 1 DÍGITO	**DECIMAL**
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

Tabla 5: Los 16 posibles valores que se pueden codificar con 4 bits

Entonces, como podemos ver en las dos tablas anteriores:

- 3 dígitos binarios equivalen a 1 dígito octal → $2^3 = 8^1 = 8$ posibles valores.
- 4 dígitos binarios equivalen a 1 dígito hexadecimal → $2^4 = 16^1 = 16$ posibles valores.

5) Intenta realizar la conversión a decimal del número octal 325.

El proceso de conversión a decimal es igual que si hiciéramos la conversión de binario a decimal, pero en este caso la base del sistema de numeración es 8 (b=8) en lugar de 2.

$\mathbf{325}_8 = \mathbf{3}\times 8^2 + \mathbf{2}\times 8^1 + \mathbf{5}\times 8^0 = 3\times 64 + 2\times 8 + 5\times 1 = 192 + 16 + 5 = \mathbf{213_{10}}$

6) ¿Serías capaz de escribir el proceso de conversión entre números en octal y binario, y viceversa? (nota: utiliza como punto de partida el proceso que se realiza entre hexadecimal y binario).

En el sistema hexadecimal dividimos el número binario a convertir en grupos de 4 bits ya que, como hemos visto en el ejercicio 4 (ver Tabla 5), con 4 bits podemos representar 16 valores diferentes; es decir, todo el alfabeto de valores hexadecimales.

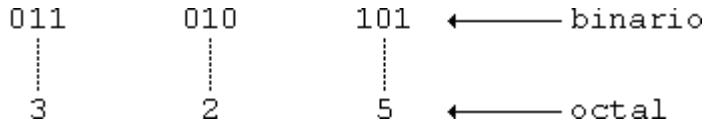
Por tanto, para la conversión de números binarios al sistema octal (8 posibles valores diferentes) podemos hacerlo dividiendo ese número en grupos de 3 bits, como hemos visto en

el ejercicio 4 (ver Tabla 4), y añadiendo al último grupo todos los ceros a la izquierda que sean necesarios.

Pongamos como ejemplo el número binario 11010101. Para su conversión al sistema de numeración octal, vamos dividiendo ese número en grupos de 3 empezando por la derecha; es decir, obtenemos 3 grupos diferentes: 11, 010 y 101. Al último grupo le añadimos un 0 a la izquierda para rellenar ese grupo hasta que tenga 3 dígitos.

Por tanto, tenemos los 3 grupos de 3 dígitos: 011, 010 y 101.

Después de esa división la conversión es directa (ver Tabla 4), obteniendo finalmente, como podemos ver en la siguiente figura, el valor 325 en el sistema octal (mismo valor que en el ejercicio 5).



La conversión de cada grupo, si no sabemos la conversión directa entre el binario y el octal, se haría de la siguiente manera:

$\mathbf{011}_2 = \mathbf{0}\times2^2 + \mathbf{1}\times2^1 + \mathbf{1}\times2^0 = 1\times2^1 + 1\times2^0 = 2 + 1 = 3_{10}$

$\mathbf{010}_2 = \mathbf{0}\times2^2 + \mathbf{1}\times2^1 + \mathbf{0}\times2^0 = 1\times2^1 = 2_{10}$

$\mathbf{101}_2 = \mathbf{1}\times2^2 + \mathbf{0}\times2^1 + \mathbf{1}\times2^0 = 1\times2^2 + 1\times2^0 = 4 + 1 = 5_{10}$

7) ¿Cuál es el siguiente número hexadecimal al 19F? a) 2A0 b) 200 c) 1A0

Si empezamos por la cifra de más a la derecha, la F, tenemos que el siguiente número es el **0** ya que la F es la última cifra del alfabeto hexadecimal. Extrapolándolo a nuestro sistema decimal, lo mismo ocurre cuando del número 9 pasamos directamente al 0, debido a que ya no existen más números en el alfabeto decimal.

Al cambiar de la F al 0 tenemos que sumar 1 (el acarreo) a la siguiente cifra; al igual que ocurre en el sistema decimal. En este caso, al 9 le sumamos 1. El resultado en hexadecimal es **A**, que correspondería con el número 10 en decimal. Como pasar del 9 a la A no supone volver al primer valor del alfabeto, la tercera cifra no sufre ninguna modificación (no hay acarreo). Por tanto se quedaría el **1**.

<u>Solución</u>: El siguiente número al $19F_{16}$ corresponde con el $1A0_{16}$ (**opción C**).

8) ¿Cuántos bytes tienen 16 MB? Y, ¿cuántos bits?

Si 1 MB corresponde a 1024 KB entonces 16 MB serán 16×1024 KB. Es decir, 16 MB corresponden a 16384 KB en total. A su vez, 1 KB corresponde a 1024 bytes; por lo que 16384 KB serán 16384×1024 bytes. Es decir, **16777216 bytes** en total.

Por otro lado, como 1 byte corresponde a 8 bits entonces 16777216 bytes serán 16777216×8 bits. Es decir, 16 MB corresponde a **134217728 bits** en total.

Resumiendo: 16 MB = 16×1024 KB = 16384 KB = 16384×1024 bytes = 16777216 bytes = 16777216×8 bits = 134217728 bits

9) Si tengo 2 módulos de memoria con 512 MB y 1024 MB, ¿cuántos GB de memoria tengo en total?

La suma total de ambos módulos sería 512 MB + 1024 MB = 1536 MB.

Por otro lado, si 1 GB corresponde a 1024 MB entonces 1536 MB serán:

1536 MB / 1024 MB = **1,5 GB**

10) Tenemos un disco duro con una capacidad total de 20 GB. Si cada bloque en los que ese disco duro se divide tiene 4 KB, ¿cuántos bloques hay en total?

Lo primero, tendremos que convertir la capacidad del disco duro de GB a KB. Recordemos que 1 GB corresponde a 1024 MB (primera conversión), por lo que 20 GB serán:

20×1024 = 20480 MB

A su vez, 1 MB equivale a 1024 KB (segunda conversión). Por tanto, si tenemos un disco duro de 20480 MB, en total tendremos también:

20480×1024 = 20971520 KB

Finalmente, si cada bloque en los que el disco duro se divide ocupa 4 KB, tendremos un total de:

20971520 KB / 4 KB = **5242880 bloques**

TABLA ASCII

El código 32 es el espacio en blanco. Los códigos del 33 al 126 se conocen como **caracteres imprimibles**, y representan letras, dígitos, signos de puntuación, etc.

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