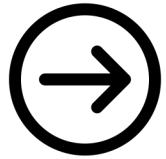
Módulo 2

Representación de Información

¿Qué lenguaje hablan las computadoras?



```
101010010
100100100
100100101
001001001
001010010
010010001
001100110
```

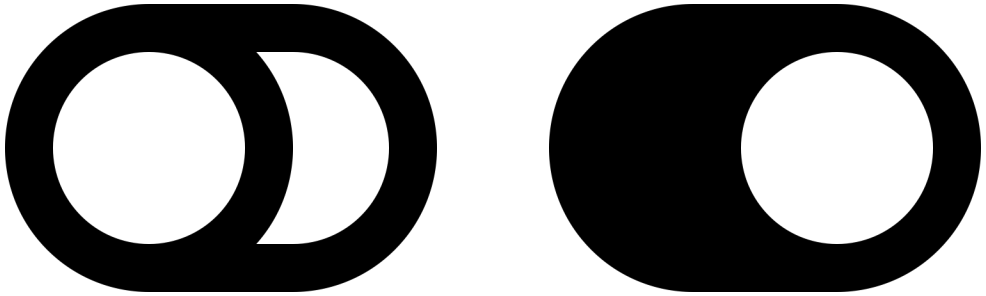
Para poder interpretar archivos, las computadoras utilizan el **lenguaje binario** basado en bits.

Bits

Binary digIT

Un bit es la unidad más básica de información en el mundo digital. Un bit puede representar únicamente dos valores: **0** ó **1**.

Estos valores se pueden interpretar también como TRUE o FALSE, o los estados **encendido** o **apagado**.



¿Por qué utilizamos los bits?

1. Los transistores manejan dos estados: encendido y apagado

2. Es más fácil trabajar con 2 posibles valores que con 3, 4, 5…

3. El almacenamiento y procesamiento en binario es muy confiable

HOW HARD DRIVES WORK

La siguiente tabla contiene los voltajes leídos de un sector de un disco duro.

0 = De 0V hasta 1.75V
1 = De 1.76V hasta 3.5V

3.4	3	0.5	0.7
3.3	3.1	0.73	0.35
0.6	0.7	0.8	2.2

0V hasta 1.75V	1.76V hasta 3.5V

3.4	3	0.5	0.7
3.3	3.1	0.73	0.35
0.6	0.7	0.8	2.2

1	1	0	0
1	1	0	0
0	0	0	1

0V hasta 1.75V	1.76V hasta 3.5V

3.4	3	0.5	0.7
3.3	3.1	0.73	0.35
0.6	0.7	1.74	2.2

1	1	0	0
1	1	0	0
0	0	0	1

Aunque la memoria sufrió una afectación externa, la información continúa consistente.

Supongamos que ahora manejamos 4 rangos:

0 = De 0V hasta 0.75V
1 = De 0.76V hasta 1.50V
2 = De 1.51 hasta 2.25V
3 = De 2.26V hasta 3.0V

1.76	2	1	0
2.5	2.2	1.2	0.9
0.8	1.6	1.74	2.2

0V hasta 0.75V	0.76V hasta 1.5V	1.51V hasta 2.25V	2.26V hasta 3.0V

1.76	2	1	0.75
2.5	1.51	1.2	0.9
0.8	1.6	1.74	2.2

2	2	1	0
3	2	1	1
1	2	2	2

0V hasta 0.75V	0.76V hasta 1.5V	1.51V hasta 2.25V	2.26V hasta 3.0V

1.76	2	1	0.75
2.5	1.51	1.2	0.9
0.8	1.6	2.26	2.2

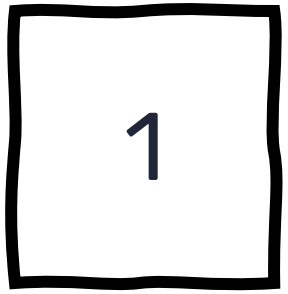
2	2	1	0
3	2	1	1
1	2	2	2

Una pequeña modificación en uno de los valores afecta el contenido de los registros. Entre mas rangos haya, mayor el riesgo de corrupción de información.

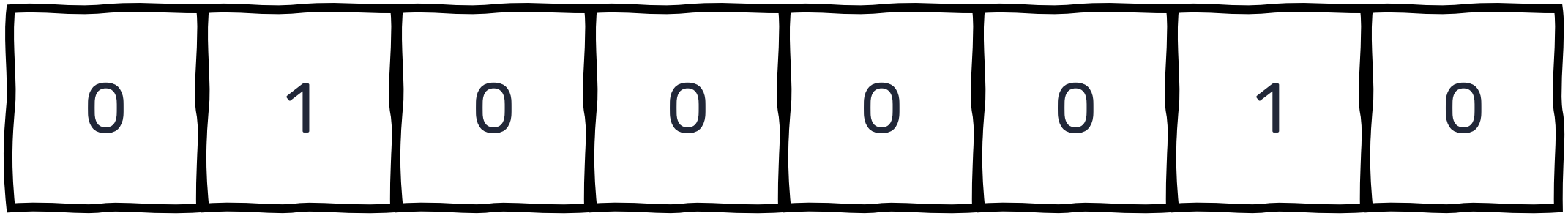
Bits y Bytes

Byte

Un bit



Un grupo de 8 bits es conocido como un **byte**.



Velocidades de Internet

¿Cómo se mide el ancho de banda que te ofrece una compañía como Telmex, Axtel, Izzi?

50 Megabits por segundo

↓ ↓ ↓ ↓

Dividimos entre 8

↓ ↓ ↓ ↓

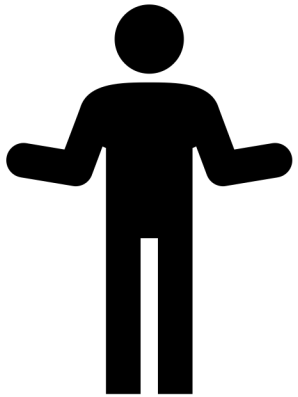
6.25 Megabytes por segundo

Métrico

Binario IEC

Binario JEDEC

La diferencia entre estos sistemas es si la memoria se mide en potencias de dos, o potencias de 10.

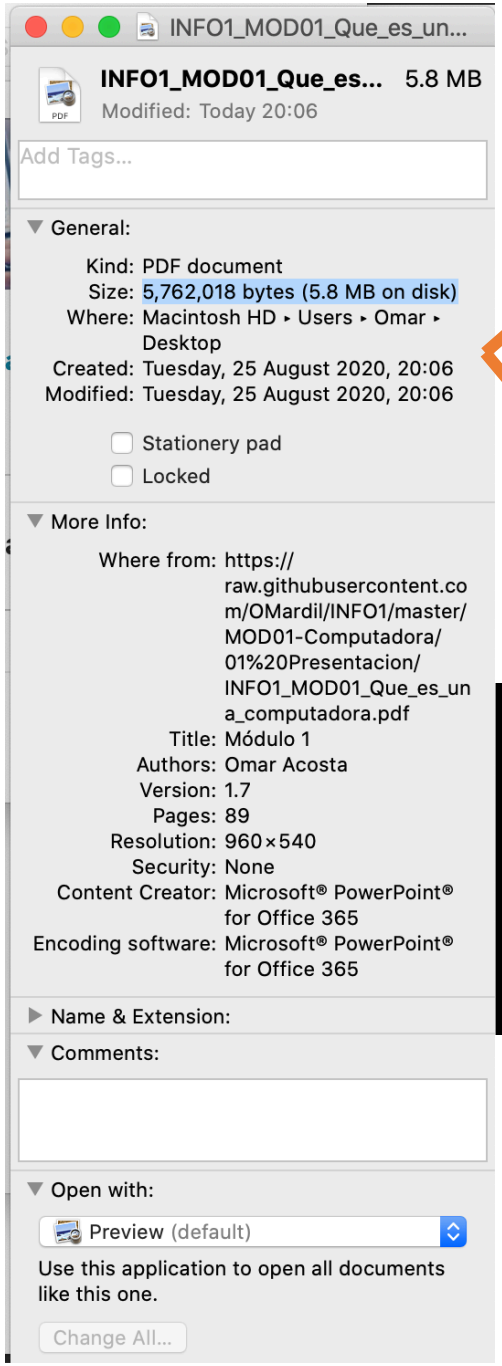


Sistema Métrico

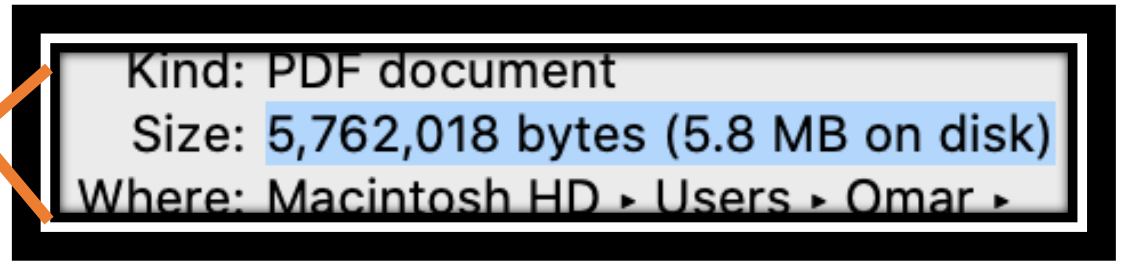
Valor (bytes)	Decimal	
1	B	byte
1000	kB	kilobyte
1000^2	MB	megabyte
1000^3	GB	gigabyte
1000^4	TB	terabyte
1000^5	PB	petabyte

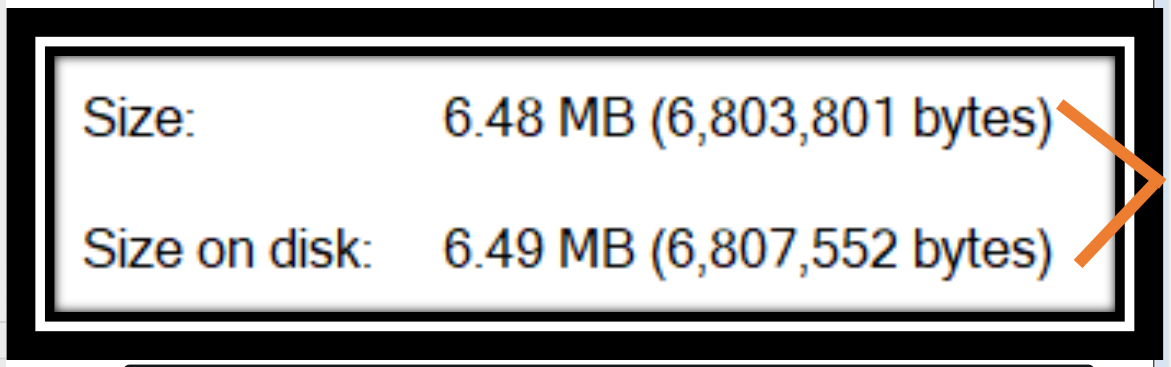
Sistema Binario

Valor (bytes)	IEC		JEDEC	
$2^0 = 1$	B	byte	B	byte
$2^{10} = 1024$	KiB	kibibyte	KB	kilobyte
2^{20}	MiB	mebibyte	MB	megabyte
2^{30}	GiB	gibibyte	GB	gigabyte
2^{40}	TiB	tebibyte	TB	terabyte
2^{50}	PiB	pebibyte	PB	petabyte

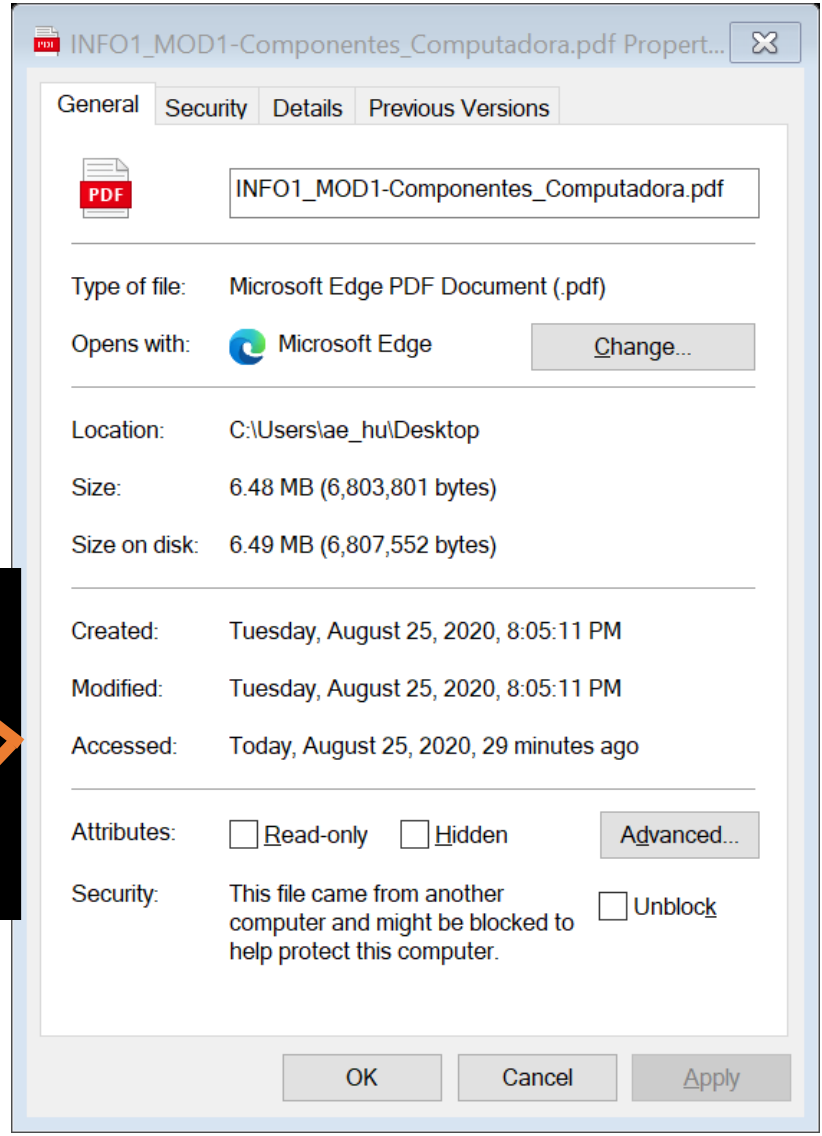


Mac OS utiliza el sistema métrico (potencias de 10) para medir los archivos.





Windows utiliza el sistema binario JEDEC (potencias de 2) para medir los archivos.



Velocidades de Internet

¿Cómo se mide el ancho de banda que te ofrece una compañía como Telmex, Axtel, Izzi?

- 50 Megabits por segundo
- 6.25 Megabytes por segundo (sistema decimal)
- 5.96046 Mebibytes por segundo (sistema IEC)
- 5.96046 Megabytes por segundo (sistema JEDEC)

What You Buy	What You Get, Base 2	What You Get, Base 10	What's Wrong?
8 Gigabytes of RAM	8 Gibibytes	8.59 Gigabytes	Sold as gigabytes, but is actually gibibytes
768 Gigabytes of RAM	768 Gibibytes	824.6 Gigabytes	Sold as gigabytes but is actually gibibytes
256 Gigabyte SD card	238.4 Gibibytes	256 Gigabytes	Sold as gigabytes, shows up in computers as Gibibytes
6 TB HDD	5.45 Tebibytes	6 Terabytes	Sold as terabytes, shows up in computers as Tebibytes

Caracteres

Caracteres

Para transmitir información escrita, los humanos utilizamos un alfabeto, o un conjunto de símbolos. Cada símbolo es denominado un **caracter**.

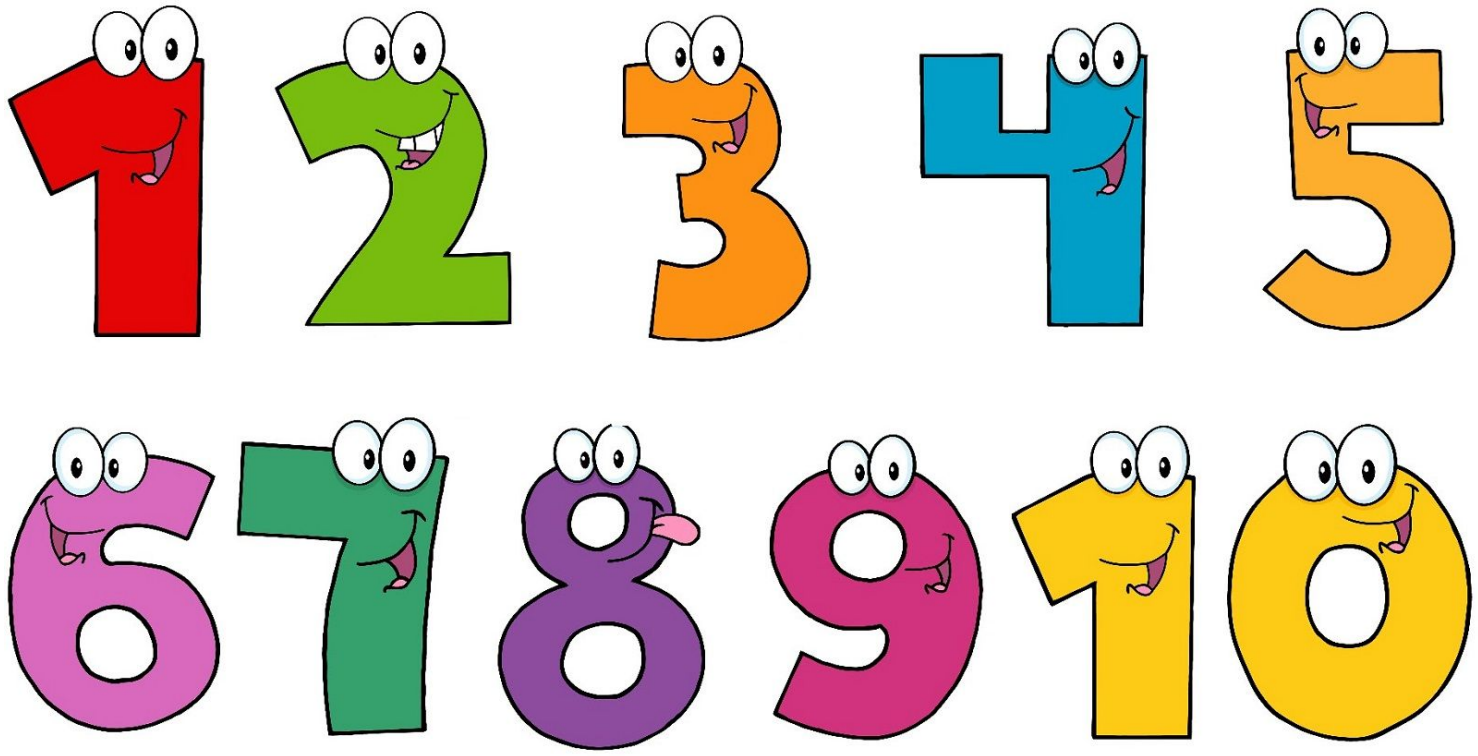
Cada caracter tiene algún significado: sonidos, pausas, números, sentimientos, etc.

Caracteres alfabéticos

Representan las 26 letras del alfabeto inglés. Desde la “a” hasta la “z”, en mayúsculas y minúsculas.

Caracteres Numéricos

Representan los números de base decimal. Desde “0” hasta el “9”.



Caracteres Especiales

Estos caracteres forman parte de alfabetos de otros lenguajes como el español, francés, alemán, etc. También incluye símbolos que indican énfasis, preguntas, enumeraciones, etc.

- SPACE
- () {} [] < >
- @
- ¡ ! ¿ ?
- , . ;
- = + - / * %
- Ñ ß á ë î

Caracteres de Control

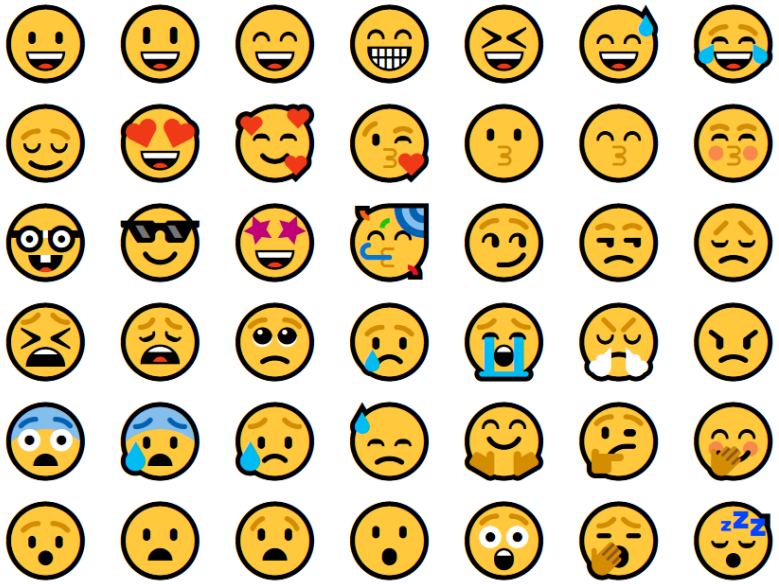
Son caracteres que no tienen representación gráfica pero sirven de control para indicar espacios tabulados, segmentos, saltos de línea, etc.

- TAB
- NULL
- BELL
- Backspace
- Line Feed
- Carriage Return

Caracteres gráficos

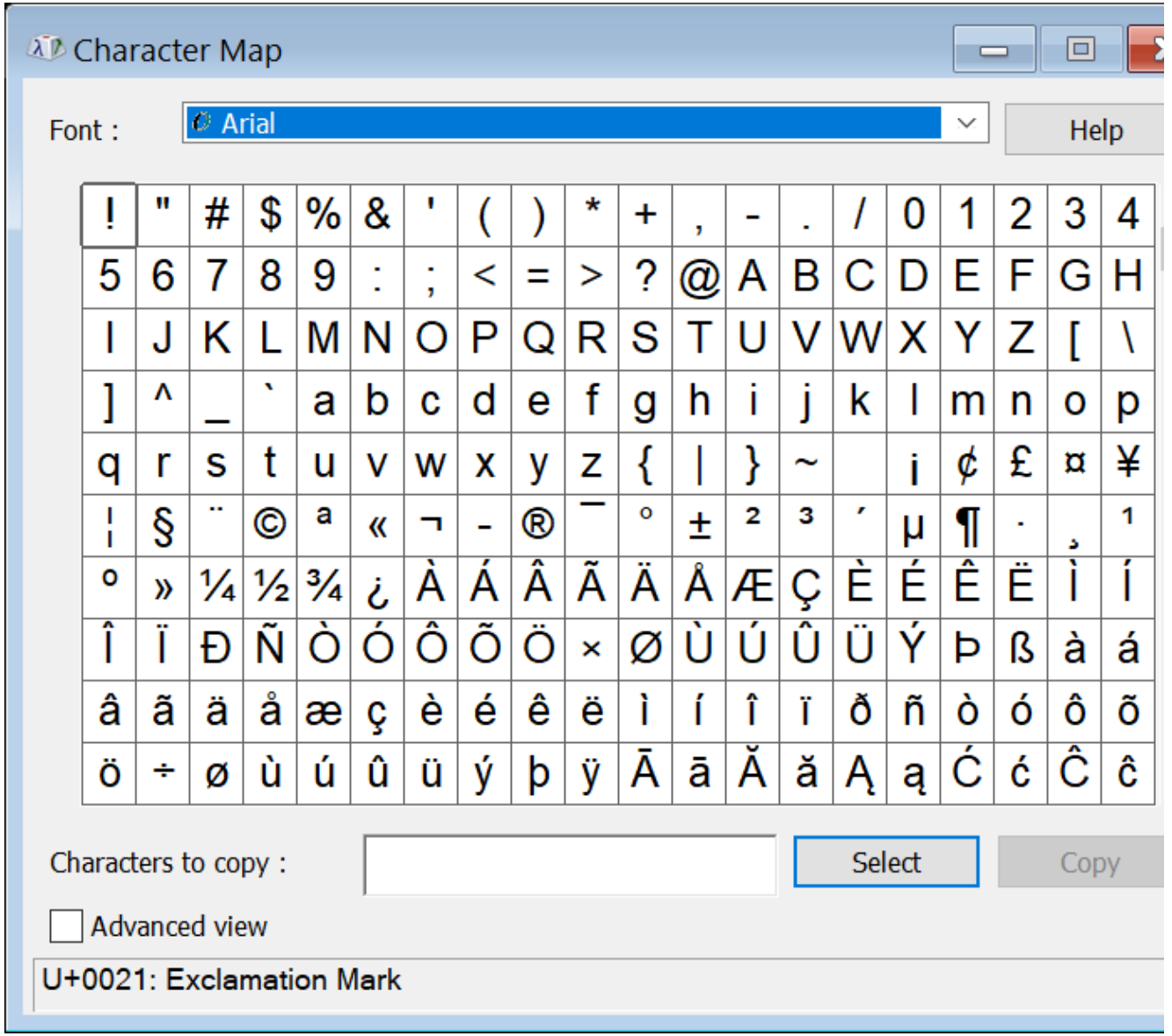
Sirven para representar caracteres de alfabetos chinos, japoneses, árabes, emojis, e íconos gráficos. También podemos incluir emojis en esta categoría.

- ® ฿ Ⓟ ☺️
- →殱☞⇏➘⛒⛄★
- Emojis



Character Map

En Windows, puedes acceder a la lista de los íconos disponibles con el programa "Character Map".



Sistema Decimal

Sistema Decimal

Para representar números, las personas utilizamos el sistema decimal.

Esto significa que cada posición de un número puede representar **10 posibles valores distintos**.

Número menor	Número Mayor
0	9

Sistema Decimal

Para indicar que estamos utilizando un sistema con base 10, podemos especificar el subíndice 10.

$$7_{10}$$
$$257_{10}$$
$$100_{10}$$

0 ← Comenzamos con un dígito, con el primer valor de la escala decimal
1
2
3
4
5
6
7
8
9 ← Llegamos al mayor valor de la escala decimal
10 ← Incrementamos los dígitos, y volvemos a empezar con el primer valor de la escala decimal
11
12
13
14
15

Sistema Binario

Sistema Binario

Las computadoras representan información utilizando el sistema binario. La base que utiliza es 2, es decir, cada dígito puede indicar únicamente **dos valores distintos**.

Número menor	Número Mayor
0	1

Sistema Binario

Para indicar que estamos utilizando numeración binaria, se especifica el subíndice 2.

$$1010010101_2$$
$$11111111_2$$
$$1010_2$$

0 ← Comenzamos con un dígito, con el primer valor de la escala binaria
1 ← Llegamos al mayor valor de la escala binaria
10 ← Incrementamos el total de dígitos a dos
11
100 ← Incrementamos el total de dígitos a tres
101
110
111
1000 ← Incrementamos el total de dígitos a cuatro
1001
1010
1011
1100
1101
1110
1111

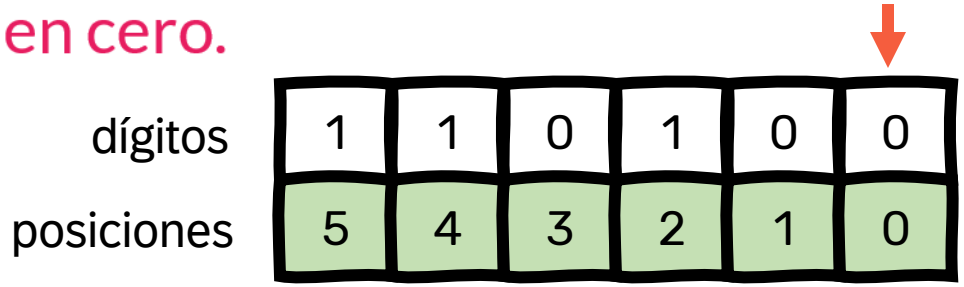
Conversiones de Binario a Decimal

Conversiones de Binario a Decimal

1. Dado un número en binario:

$$110100_2$$

2. Comenzamos numerando las posiciones de derecha a izquierda, comenzando en cero.



dígitos	1	1	0	1	0	0
posiciones	5	4	3	2	1	0

3. Sumamos la multiplicación de cada dígito binario y la base (2) elevada a la posición.

$$110100_2 = 1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0$$

$$110100_2 = 32_{10} + 16_{10} + 0_{10} + 4_{10} + 0_{10} + 0_{10}$$

$$\boxed{110100_2 = 52_{10}}$$

Otro ejemplo...

Problema 1:

$$10101011_2$$

dígitos	1	0	1	0	1	0	1	1
posiciones	7	6	5	4	3	2	1	0

$$10101011_2 = 1 \cdot 2^7 + \cancel{0 \cdot 2^6} + 1 \cdot 2^5 + \cancel{0 \cdot 2^4} + 1 \cdot 2^3 + \cancel{0 \cdot 2^2} + 1 \cdot 2^1 + 1 \cdot 2^0$$

$$10101011_2 = 1 \cdot 2^7 + 1 \cdot 2^5 + 1 \cdot 2^3 + 1 \cdot 2^1 + 1 \cdot 2^0$$

$$10101011_2 = 128_{10} + 32_{10} + 8_{10} + 2_{10} + 1_{10}$$

$$\boxed{10101011_2 = 171_{10}}$$

Conversiones de Decimal a Binario

1000_{10} :

Conversiones de Decimal a Binario

Dividimos el número a convertir entre 2, que es la base deseada

Conversiones de Decimal a Binario

1000_{10} :

$$2 \,|\, \underline{1000} \;\Big|\; 0$$

Residuo

Resultado

Apuntamos el valor del residuo.

Conversiones de Decimal a Binario

1000_{10} :

$$\begin{array}{r|l} 2\lfloor 1000 & 0 \\ 2\lfloor 500 & 0 \end{array}$$

Residuo

Resultado

Se baja el valor del resultado de la división. Se repite la operación y apuntamos el residuo.

Conversiones de Decimal a Binario

1000_{10} :

$2 \lfloor 1000$	0
$2 \lfloor 500$	0
$2 \lfloor 250$	0

Residuo

Resultado

Se baja el valor del resultado de la división. Se repite la operación y apuntamos el residuo.

Conversiones de Decimal a Binario

1000_{10} :

$2 \lfloor 1000$	0
$2 \lfloor 500$	0
$2 \lfloor 250$	0
$2 \lfloor 125$	1

Residuo

Resultado

El resultado de la división entera 125/2 = 62, con un residuo de 1. Apuntamos el residuo y el resultado de la división entera.

Conversiones de Decimal a Binario

1000_{10} :

$2 \lfloor 1000$	0
$2 \lfloor 500$	0
$2 \lfloor 250$	0
$2 \lfloor 125$	1
$2 \lfloor 62$	0

Residuo

Resultado

Repetimos la operación hasta que no podamos dividir el número.

Conversiones de Decimal a Binario

1000_{10} :

$2 \lfloor 1000$	0
$2 \lfloor 500$	0
$2 \lfloor 250$	0
$2 \lfloor 125$	1
$2 \lfloor 62$	0
$2 \lfloor 31$	1

Residuo

Resultado

Repetimos la operación hasta que no podamos dividir el número.

Conversiones de Decimal a Binario

1000_{10} :

$2 \lfloor 1000$	0
$2 \lfloor 500$	0
$2 \lfloor 250$	0
$2 \lfloor 125$	1
$2 \lfloor 62$	0
$2 \lfloor 31$	1
$2 \lfloor 15$	1

Residuo

Resultado

Repetimos la operación hasta que no podamos dividir el número.

Conversiones de Decimal a Binario

1000_{10} :

$2 \lfloor 1000$	0
$2 \lfloor 500$	0
$2 \lfloor 250$	0
$2 \lfloor 125$	1
$2 \lfloor 62$	0
$2 \lfloor 31$	1
$2 \lfloor 15$	1
$2 \lfloor 7$	1

Residuo

Resultado

Repetimos la operación hasta que no podamos dividir el número.

Conversiones de Decimal a Binario

1000_{10} :

$2 \lfloor 1000$	0
$2 \lfloor 500$	0
$2 \lfloor 250$	0
$2 \lfloor 125$	1
$2 \lfloor 62$	0
$2 \lfloor 31$	1
$2 \lfloor 15$	1
$2 \lfloor 7$	1
$2 \lfloor 3$	1

Residuo

Repetimos la operación hasta que no podamos dividir el número.

Resultado

Conversiones de Decimal a Binario

1000_{10} :

$2 \lfloor 1000$	0
$2 \lfloor 500$	0
$2 \lfloor 250$	0
$2 \lfloor 125$	1
$2 \lfloor 62$	0
$2 \lfloor 31$	1
$2 \lfloor 15$	1
$2 \lfloor 7$	1
$2 \lfloor 3$	1
$2 \lfloor 1$	1
0	

Residuo

Resultado es 0!

La división entera 1/2 = 0, con un residuo de 1. Apuntamos el 1 y concluimos.

Conversiones de Decimal a Binario

1000_{10} :

$$
\begin{array}{r|l}
2\lfloor 1000 & 0 \\
2\lfloor 500 & 0 \\
2\lfloor 250 & 0 \\
2\lfloor 125 & 1 \\
2\lfloor 62 & 0 \\
2\lfloor 31 & 1 \\
2\lfloor 15 & 1 \\
2\lfloor 7 & 1 \\
2\lfloor 3 & 1 \\
2\lfloor 1 & 1 \\
0 &
\end{array}
\Bigg\} = 1111101000_2
$$

Para finalizar, copiamos los números de abajo hacia arriba.

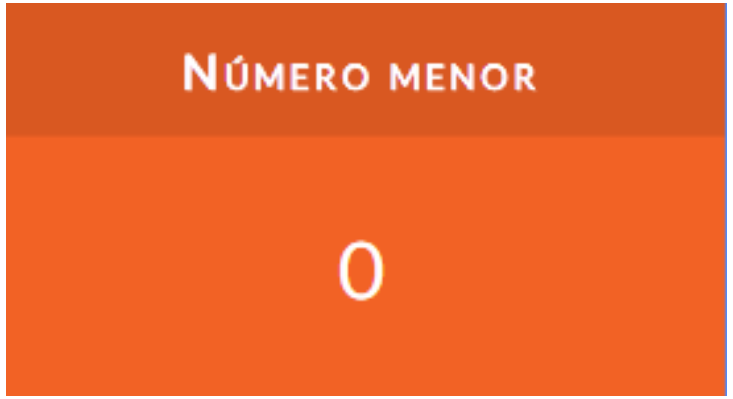
Trabajo individual

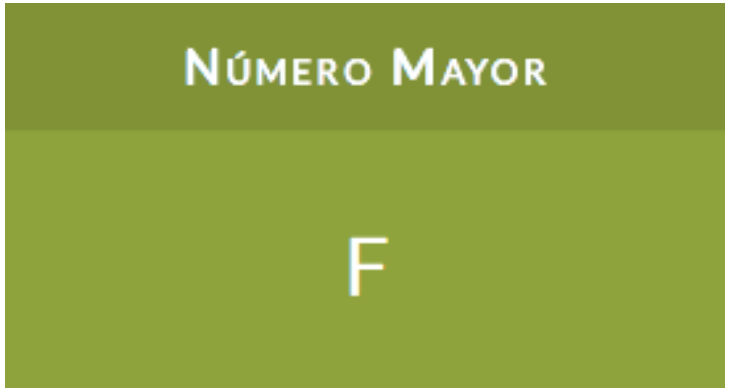
$$100001_2 \rightarrow x_{10}$$
$$11111111_2 \rightarrow x_{10}$$
$$254_{10} \rightarrow x_2$$
$$17_{10} \rightarrow x_2$$

Sistema Hexadecimal

Sistema Hexadecimal

En un sistema hexadecimal, cada dígito puede representar 16 distintos valores. Para esto se utilizan los números del 0 → 9, y posteriormente las letras A → F.





¿En donde usamos el hexadecimal?

Al representar colores en formato RGB se utiliza hexadecimal para cada valor.

En redes, las direcciones MAC se representan en hexadecimal

MAC Address
00-17-FC-34-00-00
00-17-FC-25-00-00
00-17-FC-11-00-00
00-17-FC-72-00-00
00-17-FC-80-00-00
00-17-FC-31-00-00
00-17-FC-90-00-00
00-17-FC-41-00-00
00-17-FC-51-00-00
00-17-FC-61-00-00

Sistema Hexadecimal

En las ciencias computacionales, la base hexadecimal la vemos representada con el prefijo “**`0x`**”.

- **`0xF3`**
- **`0x6A23`**
- **`0x1745`**

Otra representación utilizada es indicar la base:

- $F3_{16}$
- $6A23_{16}$
- 1745_{16}

0	E	1C
1	F ← Máximo valor 1 dígito	1D
2	10	1E
3	11	1F
4	12	20
5	13	...
6	14	FE
7	15	FF ← Máximo valor 2 dígitos
8	16	100
9	17	
A	18	
B	19	
C	1A	
D	1B	

Binario a Hexadecimal

Dado el siguiente número binario:

$$110000101101_2 \rightarrow x_{16}$$

1. De derecha a izquierda, agrupamos grupos de 4 bits.

$$\boxed{1100}\boxed{0010}\boxed{1101}_2 \rightarrow x_{16}$$

2. Convertimos cada sección a hexadecimal. Recuerda que cada posición puede tener valores del 0 al F.

$$\boxed{1100}_2 = 12_{10} = C_{16}$$
$$\boxed{0010}_2 = 2_{10} = 2_{16}$$
$$\boxed{1101}_2 = 13_{10} = D_{16}$$

3. Juntamos los números.

$$110000101101_2 = C2D_{16}$$

Conversiones de Hexadecimal a Decimal

Dado un número en hexadecimal:

$$23E_{16}$$

1. Extendemos el polinonio a través de multiplicar cada dígito *convertido a decimal* por la base origen elevado a su posición.

$$\text{Recordemos que: } E_{16} = 14_{10}$$

$$23E_{16} = 2 \cdot 16^2 + 3 \cdot 16^1 + 14 \cdot 16^0$$

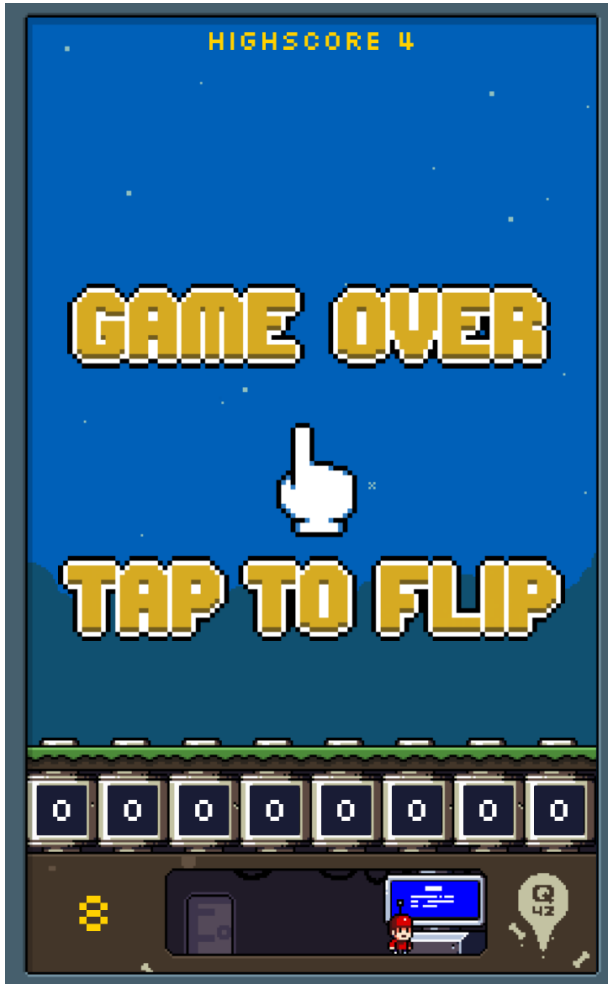
$$23E_{16} = 512_{10} + 48_{10} + 14_{10}$$

2. Completando la suma

$$23E_{16} = 574_{10}$$

Práctica Individual

https://flippybitandtheattackofthehexadecimalsfrombase16.com/



iOS

Android

Codificación

Codificación de Información

Podemos expresar la cantidad de valores representanbles “M” en función de la cantidad de bits “n” con la siguiente relación:

$$M(n) = 2^n$$

Por ejemplo, con un byte (8 bits) se pueden representar:

$$M(8) = 2^8 = 256$$

Codificación

Podemos despejar la ecuación para obtener la cantidad mínima para representar un grupo de valores, por ejemplo:

$$n \geq \log_2(M)$$
$$n \geq 3.32 \cdot \log_{10}(M)$$

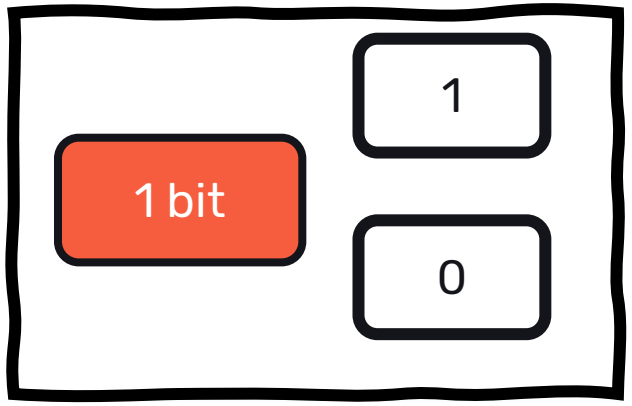
Por ejemplo, para codificar el siguiente alfabeto:

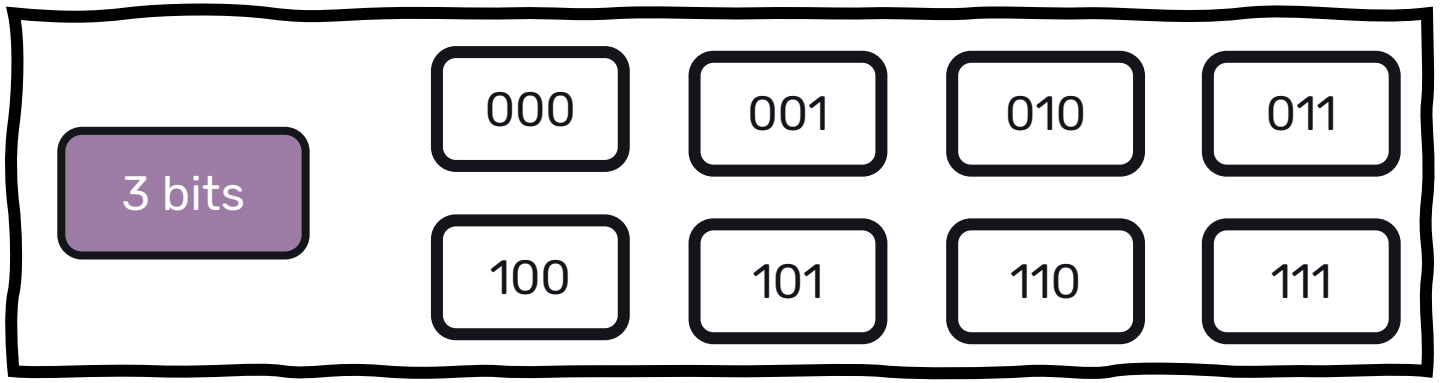
$$\{0,1,2,3,4,5,6,7,8,9\}$$

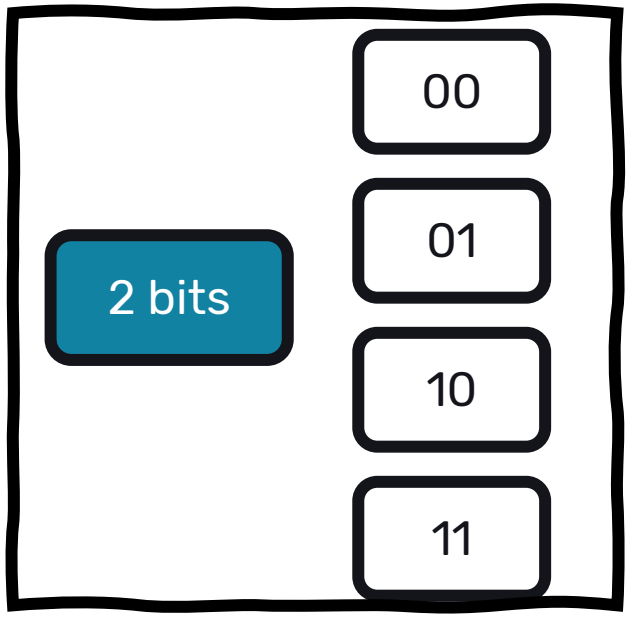
Se necesitan:

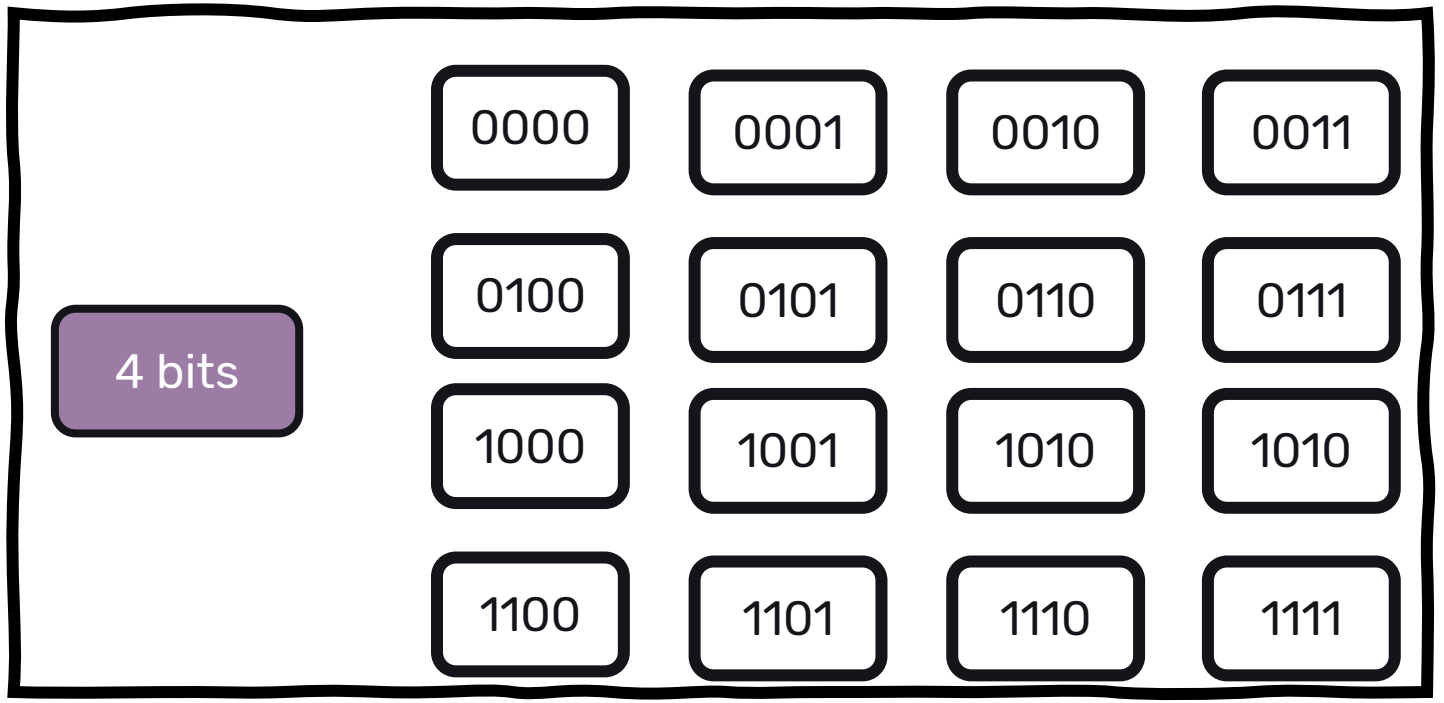
$$n \geq 3.32 \cdot \log(10)$$
$$n \geq 3.32$$
$$n \geq 4$$

Otra forma...


1 bit
1
0


3 bits
000
001
010
011
100
101
110
111

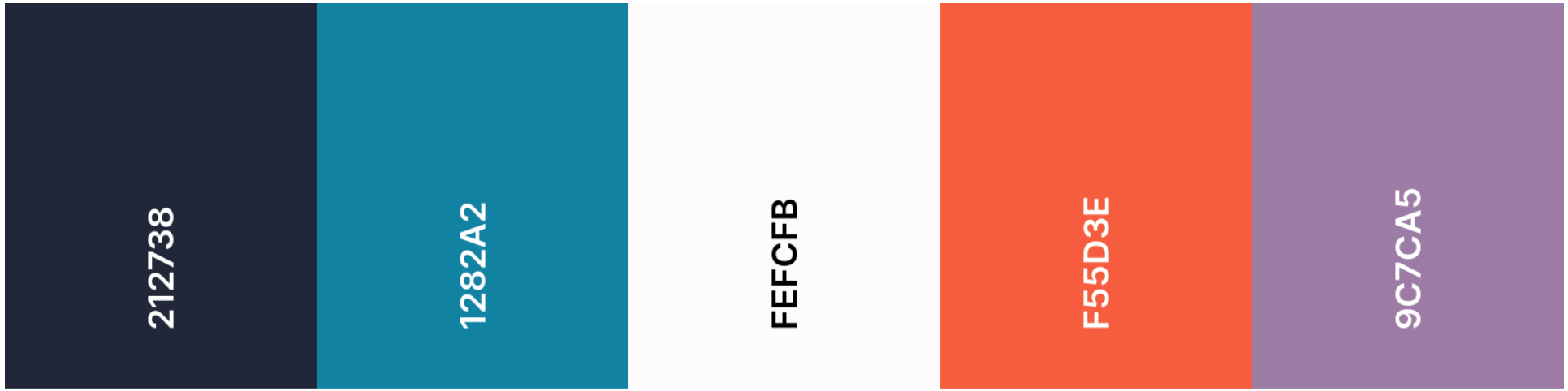

2 bits
00
01
10
11


4 bits
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1010
1100
1101
1110
1111

Ejemplo

Entonces si queremos representar un alfabeto de 700 posibles valores, podemos ir incrementando la cantidad de bits que excedan o iguale 700.

1 bit = 2^1 = 2 valores
2 bits = 2^2 = 4 valores
3 bits = 2^3 = 8 valores
4 bits = 2^4 = 16 valores
5 bits = 2^5 = 32 valores
6 bits = 2^6 = 64 valores
7 bits = 2^7 = 128 valores
8 bits = 2^8 = 256 valores
9 bits = 2^9 = 512 valores
10 bits = 2^{10} = 1024 valores



Modulo 1

coolors