Capítulo 2. Sistemas numéricos

Elsa Ortega de Ávila[1]
Néstor Alejandro Carrillo López[2]

DOI: https://doi.org/10.52501/cc.440.02

Resumen

Los sistemas numéricos surgieron en las antiguas civilizaciones como respuesta a la necesidad de contar, medir y registrar información en actividades como el comercio, la agricultura y la construcción. A lo largo del tiempo, diversas culturas desarrollaron sistemas adaptados a sus contextos. Entre ellos destacan el sistema decimal, utilizado por los egipcios y perfeccionado en la India con la notación posicional; el sistema babilónico de base 60, aún vigente en la medición del tiempo y los ángulos; el sistema egipcio jeroglífico; y el sistema romano, limitado para cálculos complejos. Asimismo, el sistema maya introdujo el concepto de cero en una base vigesimal, mientras que el sistema binario, formalizado por Leibniz, se convirtió en la base de la informática moderna. El sistema indoarábigo, con su notación posicional y uso del cero, se consolidó como el más eficiente y universal.

En la actualidad, los sistemas numéricos se definen como estructuras que permiten representar cantidades y realizar operaciones, diferenciándose por su base y número de símbolos; los principales son el decimal (base 10), el binario (base 2), el octal (base 8) y el hexadecimal (base 16), estos últimos ampliamente utilizados en informática. Además, es posible convertir números entre distintos sistemas mediante métodos específicos, generalmente usando el sistema binario como intermediario. Estas conversiones implican procesos como la agrupación de dígitos y el uso de potencias o divisiones sucesivas. En conjunto, los sistemas numéricos constituyen un elemento esencial para las matemáticas, la ciencia y la tecnología.

[1] Doctora en Ciencias de la Educación. Docente en Tecnológico Nacional de México. ORCID: https://orcid.org/0000-0003-0405-2978 ; correo electrónico: elsa.ortega@itz.edu.mx
[2] Maestro en Informática Administrativa. Encargado del laboratorio de Manufactura en Tecnológico Nacional de México.

Palabras clave: *sistemas numéricos, binario, octal, hexadecimal, decimal.*

Sistemas históricos y contemporáneos

Los sistemas numéricos tienen su origen en las antiguas civilizaciones, puesto que necesitaban contar, medir y registrar información para actividades prácticas como el comercio, la agricultura y la construcción. Con el crecimiento de las sociedades y la complejidad de sus actividades, surgió la necesidad de desarrollar métodos organizados y eficientes para representar números (Ifrah, 2000).

1. Sistema decimal (base 10). Civilización mesopotámica y egipcia

Uno de los primeros sistemas numéricos adoptados ampliamente fue el decimal, basado en diez unidades, posiblemente porque los humanos tenemos diez dedos, lo que facilita contar con ellos. Los antiguos egipcios y babilonios usaron este sistema para representar cantidades en sus registros (Menninger, 1992). Más adelante, el sistema decimal fue perfeccionado en India y transmitido al mundo occidental por matemáticos árabes, quienes desarrollaron la notación posicional y promovieron el uso del sistema decimal como base para los cálculos (Joseph, 2011).

2. Sistema babilónico (base 60)

Hace más de 4000 años, los babilonios de la antigua Mesopotamia desarrollaron un sistema numérico sexagesimal (base 60) que se sigue usando hoy en día para medir el tiempo (minutos y segundos) y los ángulos (Andonegui, 2007; Ifrah, 2000). Este sistema combinaba características de las bases 10 y 60, lo cual demuestra una sofisticación notable en el desarrollo de sistemas numéricos (Robson, 2008).

3. Sistema egipcio (base 10 con jeroglíficos)

Los egipcios empleaban un sistema decimal que utilizaba jeroglíficos específicos para representar valores como 1, 10, 100, y así sucesivamente, sumando estos símbolos para formar otros números (Clagett, 1989). Este sistema carecía de un cero y se usaba principalmente en registros administrativos y de contabilidad, en un formato adecuado para sus necesidades prácticas (Ifrah, 2000).

4. Sistema romano (números romanos)

Desarrollado en la antigua Roma, el sistema romano utiliza letras como “I”, “V”, “X”, “L”, “C”, “D” y “M” para representar valores y sigue un sistema aditivo y sustractivo. Este sistema carecía de un cero y era menos adecuado para cálculos matemáticos complejos, sin embargo, se mantuvo en uso durante la Edad Media en Europa (Menninger, 1992).

5. Sistema maya y mesoamericano (base 20)

Los mayas desarrollaron un sistema vigesimal (base 20) que incorpora el concepto de cero, uno de los primeros ejemplos en la historia de su uso, lo cual les permitió avances significativos en astronomía y en el desarrollo de un calendario preciso (Closs, 1986; Ifrah, 2000). Otros pueblos mesoamericanos utilizaron sistemas similares basados en la base 20, pero adaptados a sus necesidades culturales y prácticas.

6. Sistema binario (base 2). China y Europa moderna

Aunque el sistema binario moderno fue formalmente desarrollado en Europa por el matemático Gottfried Wilhelm Leibniz en el siglo XVII, sus orígenes se encuentran en el antiguo texto chino *I Ching*, que emplea secuencias binarias para representar conceptos (Gardner, 1983). El sistema binario es la base de la informática moderna, ya que permite representar datos y realizar operaciones en términos de 0 y 1 (Ifrah, 2000).

7. Sistema indoarábigo (números arábigos - base 10)

Desarrollado en la India antigua y difundido por matemáticos árabes, el sistema indoarábigo incluyó el concepto de cero y una notación posicional que facilitó cálculos complejos. Este sistema se consolidó como la base de la aritmética moderna y se usa en todo el mundo por su simplicidad y eficiencia (Joseph, 2011).

Los sistemas numéricos tienen diversos orígenes según las necesidades y contextos de cada civilización. Sin embargo, sistemas como el decimal y el indoarábigo se difundieron ampliamente, demostrando ser fundamentales en matemáticas, ciencia e ingeniería hasta hoy. En la página de la Universidad autónoma de Ciudad Juárez (Robledo et al., s.f.), comparten que:

Los sistemas de numeración permiten representar cantidades mediante símbolos organizados de acuerdo con reglas específicas de valor posicional. En estos sistemas, el valor de cada cifra depende tanto del símbolo utilizado como de la posición que ocupa dentro del número, la cual está determinada por una base numérica. De esta manera, el sistema decimal utiliza diez símbolos fundamentales (0–9), mientras que otros sistemas, como el binario o hexadecimal, emplean diferentes bases para representar información y realizar operaciones matemáticas (Rosen, 2012, p. 111).

De los sistemas mencionados, destacan y se siguen utilizando los sistemas de valor posicional, donde todos se refieren a una base específica de numeración (Andonegui, 2007).

Definiciones de los sistemas contemporáneos

Los sistemas numéricos son estructuras fundamentales en matemáticas que permiten representar cantidades y realizar operaciones aritméticas. De acuerdo con la Universidad Nacional a Distancia, los sistemas se diferencian por el número de símbolos que permiten, por ejemplo, el sistema binario consta de dos dígitos: el cero y el uno; el octal consta de ocho dígitos; el decimal, de diez dígitos; y el hexadecimal, de dieciséis dígitos (UNAD, s.f.). A continuación, se describen los principales sistemas numéricos.

Sistema decimal (base 10)

El sistema decimal es el más utilizado en la vida cotidiana. Está compuesto por diez dígitos (0, 1, 2, 3, 4, 5, 6, 7, 8 y 9) y es un sistema posicional, lo que significa que el valor de cada dígito depende de su posición en el número (Katz, 2009).

Sistema binario (base 2)

El sistema binario es esencial en informática y electrónica. Solo utiliza dos dígitos (0 y 1), y también es un sistema posicional; cada posición representa una potencia de 2 (Knuth, 1997).

Sistema octal (base 8)

El sistema octal emplea ocho dígitos (0 a 7). Aunque no es tan común en el uso diario, ha sido importante en ciertas áreas de la informática, especialmente en la programación de sistemas antiguos (Tanenbaum, 2016).

Sistema hexadecimal (base 16)

El sistema hexadecimal utiliza 16 símbolos: los dígitos del 0 al 9 y las letras A, B, C, D, E y F, que representan los valores 10, 11, 12, 13, 14 y 15, respectivamente. Es muy utilizado en programación y desarrollo de *software* (Hennessy y Patterson, 2017).

Los sistemas octales y hexadecimales

Ejercicio 1. Hacer los números del 1 al 120 en el sistema octal.

1	2	3	4	5	6	7	10
11	12	13	14	15	16	17	20

Ejercicio 2. Completar los números del 1 al 120 de los siguientes hexadecimales.

1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20
21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F	30

Conversiones entre los sistemas numéricos

Los números se pueden convertir de un sistema a otro, como las monedas de cambio; en seguida se muestra una tabla de los métodos para convertirlos. Por *default* se recomienda el sistema binario como sistema de cambio.

Binario	Octal	Decimal	Hexadecimal
Si tengo el número en binario.	Se agarran los números de 3 en 3 del número en binario y se convierte a octal en individual.	Se toma el número que le corresponde por su posición de la tabla de potencias binarias y se suma.	Se toman de a 4 números y se convierte a símbolo hexadecimal.
Se despliega de cada número octal a 3 números de binario.	**Si se tiene el número en octal, primero se convierte a binario.**	Se toma el número binario y se pone en la tabla para calcular su valor.	Se agarra el número en binario y se agarran de 4 en 4.
Se puede hacer por tabla o por división. Por división se divide entre 2 y se va guardando el residuo, ya que el residuo es el resultado, como se muestra en el método 7 de conversión.	Se agarra del binario de 3 en tres, como se muestra en el método binario a octal.	**Si se tiene el número en decimal, primero se convierte a binario.**	Se agarra del binario de 4 en cuatro como se muestra en el método binario a hexadecimal.
Se despliegan de cada número hexadecimal cuatro dígitos binarios, de acuerdo al símbolo que se tenga como se muestra en la tabla del método 3.	Del número binario se toman de 3 en 3, y se saca el número en octal como se muestra en la tabla del método 1.	Se convierte el binario a decimal, recomendable por método de tabla.	**Si se tiene el número en hexadecimal, primero se convierte a binario. Método inverso de binario a hexadecimal.**

Métodos de conversión

1. **De binario a octal.** Se toman 3 números de los binarios y se sustituyen por el número que les corresponde según la tabla siguiente:

En binario	En octal
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Por lo tanto, el número 010100 corresponderá al número 24 en octal.

2. **De binario a decimal**. Se toma un número de la tabla de potencia binaria y se pone en la tercera fila, de derecha a izquierda, como se muestra a continuación (en morado); luego se deben multiplicar los 1 por el número en rojo que le corresponde y, al final, hay que sumar todos los valores resultantes.

2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1024	512	256	128	64	32	16	8	4	2	1
			1	0	1	0	1	0	1	0
			128	+	32	+	8	+	2	

El ejemplo de la tabla anterior corresponde al número 170 decimal.

3. **De binario a hexadecimal**. Se toman los dígitos binarios de 4 en 4 y se sustituye el número de acuerdo con la siguiente tabla:

En binario	En octal
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Por ejemplo, el número 00110101 en binario, al tomarlo de 4 en 4 nos da 0011, que equivale a 3, y 0101, que equivale a 5, por lo que el número resultante en hexadecimal es 35.

4. **De octal a binario**. Se toma en cuenta la tabla para cambiarlo. Por ejemplo, si tengo 734 se despliega cada número: 7 equivale 111, 3 equivale 011 y 4 equivale a 100. Por lo tanto queda así: 111 011 100.

En octal	En binario
1	001
2	010
3	011
4	100
5	101
6	110
7	111

5. **De octal a decimal**. Primero se pasa a binario y luego se aplica el método de binario a decimal, mediante la tabla que se muestra en el método 7.

6. **De octal a hexadecimal**. Primero se pasa a binario y luego de binario a hexadecimal, como se explica en el punto 3.

7. **De decimal a binario**. Se puede hacer de 2 formas: por tabla o por división.

a. Por tabla. Se coloca el número en la tercera línea, se va poniendo un 1 en los números que sumados den el número exacto. En la tabla contigua se pone un ejemplo del número 423:

Pasos:

1. Primero se elige un número menor o igual al número 423, en este caso sería el 256, entonces pongo un 1 debajo del 256; como no es igual al 423 sigo al paso 2.
2. Después le sumo el número inmediato a la derecha, si no me paso de 423 pongo el 1 y se suma el 256 al 128 = 384, como es menor a 423, vuelvo a repetir el paso 2.
3. Se repite el paso 2 hasta que la suma de los números que contengan 1 nos den 423.
4. En el ejemplo se sumaron 256 + 128 + 32 + 4 + 2 + 1 = 423

211	210	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
2048	1024	512	256	128	64	32	16	8	4	2	1
			1	1	0	1	0	0	1	1	1
			256	128		32			4	2	1

b. **Por división** se divide el número entre la base, en este caso se divide entre 2, y se anota el residuo, si es 1 o es 0.

Número	Entre	Toca a:	Y sobra
423	entre 2	211	1
211	entre 2	105	1
105	entre 2	52	1
52	entre 2	26	0
26	entre 2	13	0
13	entre 2	6	1
6	entre 2	3	0
3	entre 2	1	1
1	entre 2	0	1

Luego se toma la columna de lo que sobra, de abajo hacia arriba, y nos queda: 110100111.

Ejercicio 3. Convertir el número 1000 en cada columna a los otros sistemas numéricos.

Binario	Octal	Decimal	Hexadecimal
1000			
	1000		
		1000	
			1000

Ejercicio 4. Completar los espacios en blanco convirtiendo entre sistemas numéricos.

Binario	Octal	Decimal	Hexadecimal
1111000			
	734		
		823	
			FF

Operaciones de los sistemas numéricos

Operaciones binarias

Primero vamos a ver las operaciones con binarios, para lo cual utilizaremos lo siguiente.

Suma en binario

1 + 0 =1

0 + 1 =1

0 + 0 = 0

1 + 1 =10

La suma de 1+1 sabemos que es 2, debe escribirse en binario con dos cifras (10), por tanto, 1+1 es 0 y se arrastra una unidad, que se suma a la posición continua hacia la izquierda. En la siguiente tabla se muestran siete ejercicios, uno por cada columna.

1	2	3	4	5	6	7
10101	111	110 0 1 1	1 1 1 0 0 0	1 1 1 1 1	1 0 0 1 1 1	1 1 1 1 1
11100	110	111 0 0 0	1 0 1 0 1 0	1 0 1 0 1	1 0 1 0 1 0	1 1 0 1 1
110001	110	1101 0 1 1	1 1 1 1 1 1	1 0 0 1 0	1 1 0 0 1 1	1 0 1 0 1
	10011		1 0 1 0 0 0 0 1	100 0 1 1 0	100 0 0 1 0 0	100 1 1 1 1

Ejemplos. Los números rojos son los resultados de las sumas.

Restas con binarios

Reglas

1 - 1	1 - 0	1 0 - 1	0 - 0
0	1	1	0

El número de arriba debe ser mayor que el de abajo.

Ejemplos. El resultado es la tercer línea.

1	2	3	4	5
1 1 0 0	1 1 0 1 1	1 1 1 1 1	1 1 0 0 1 1	1 1 1 0 0 0
1 0 1	1 0 1 0 1	1 1 0 1	0 1 1 0 1	0 0 1 0 1 1
1 1 1	**0 0 1 1 0**	**1 0 0 1 0**	**1 0 0 1 1 0**	**1 0 1 1 0 1**

Multiplicaciones con binarios

Para las multiplicaciones es muy fácil, ya que cero por cualquier número te da cero y uno por cualquier número te da el mismo número: 1 x 0 = 0 y 1 x 1 = 1.

Realicemos a continuación algunos ejercicios.

1	2	3	4	5
11101 x110	111x 110	111 0 1 1x1110	1 1 1 0 0 0x1010	1 1 1 1 1 x 1 0 1 0 1
0 0 0 0 0	0 0 0	0 0 0 0 0 0	0 0 0 0 0 0	1 1 1 1 1
1 1 1 0 1	1 1 1	1 1 1 0 1 1	1 1 1 0 0 0	0 0 0 0 0
1 1 1 0 1	1 1 1	1 1 1 0 1 1	0 0 0 0 0 0	1 1 1 1 1
1 0 1 0 1 1 1 0	1 0 1 0 1 0	1 1 1 0 1 1	1 1 1 0 0 0	0 0 0 0 0
		1 1 0 0 1 1 1 0 1 0	1 0 0 0 1 1 0 0 0 0	1 1 1 1 1
				1 0 1 0 0 0 1 0 1 1

División de binarios

Para dividir primero debemos verificar si el dividendo es mayor que el divisor, si es así continuamos.

```
      0 1 1 1
   ┌──────────
11 │ 1 0 1 1 0
     1 1
     ─────
       1 0 1
         1 1
       ─────
         1 0 0
           1 1
         ─────
             1
```

Supongamos que queremos dividir el número binario **10110** (22 en decimal) entre **11** (3 en decimal). El proceso sería el siguiente:

1. Dividimos el primer grupo de bits que sea igual o mayor al divisor:
   - **101** (equivale a 5 en decimal, mayor que 3).
2. **101** dividido entre **11** es igual a **1** con un residuo de **10**.
3. Bajamos el siguiente bit (el **1**), formando **101** (equivale a 5 en decimal).

4. Repetimos el proceso:
   - **101** entre **11** es **1** con un residuo de **10**.
5. Bajamos el siguiente bit (**0**), formando **100** (equivale a 4 en decimal).
6. **100** dividido entre **11** es igual a **1** con un residuo de **1**.
7. Como no hay más bits para bajar, hemos terminado.

Ejemplos:

```
      0 1  1 0 0
  ---------------
111 | 1 0 1  1  0  0 1
      1 1 1
  ---------------
      1 0 0 0
        1 1  1
  ---------------
        0 0 0 1  0 1
  ---------------
```

```
        1 0 0 1 1 0 0
  -------------------
101 | 1 0 1 1 1 1 1 1
      1 0 1
  -------------------
      0 0 0 1 1 1
          1 0 1
  -------------------
          0 1 0 1
            1 0 1
  -------------------
            0 0 0 1 1
  -------------------
```

```
      1 0 1 1 0 1
  ---------------
1 0 | 1 0 1 1 0 1 1
      1 0
  ---------------
      0 0 1 1
        1 0
  ---------------
        0 1 0
          1 0
  ---------------
          0 0 1 1
              1 0
  ---------------
              0 1
  ---------------
```

Operaciones octales

Suma octal

Para realizar sumas octales necesitamos la siguiente tabla:

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

Entonces lo que se hace es que si quiero sumar 2 + 3, busco la intersección: fila 2 y columna 3 y ese es el resultado.

Ejemplo. Se suman 3 + 5 y si vemos en la tabla de suma octal es 10, entonces se pone el 0 y se acarrea 1; luego 3 + 1 de acarreo son 4, más 4 nos vuelve a dar 10, por lo que se vuelve a poner el 0 y se acarrea 1; luego 7 + 1 de acarreo es 10, esto más 2 es igual a 12, se pone el 2 y se acarrea el 1; luego 2 + 1 de acarreo da 3, más 3 nos da 6.

$$\begin{array}{r} 2\,7\,3\,3 \\ 3\,2\,4\,5 \\ \hline \mathbf{6\,2\,0\,0} \end{array}$$

En la tabla siguiente se muestran cinco ejercicios de sumas

Ej. 1	Ej.2	Ej.3	Ej.4	Ej. 5
734	533	222	127	437
527	247	777	276	717
1463	**1002**	**1221**	**425**	**1356**

Restas con octales

Primero debemos verificar que el número de arriba sea mayor que el de abajo, si no lo es intercambiamos posiciones, como se observa más adelante en el ejemplo, y luego procedemos a restar los números. En el ejemplo tenemos 4 menos 7, puesto que el 4 es más pequeño debe pedirle prestado al 5 (disminuyéndolo a 4) y se convierte en 14; después buscamos el número 14 en la tabla de suma octal, precisamente en la fila 7, y lo encontramos en la columna 5, ese número es el que se pone abajo; finalmente a 4 le quitamos 2 y nos quedan 2.

$$\begin{array}{r} 27 \\ 54 \\ \hline \end{array} \qquad \begin{array}{r} 5\,4 \\ 2\,7 \\ \hline \mathbf{2\ \ 5} \end{array}$$

Realicemos algunos ejemplos:

Ejemplo 1	Ejemplo 2	Ejemplo 3	Ejemplo 4	Ejemplo 5	Ejemplo 6	Ejemplo 7
734	533	703	276	716	100	221
527	247	576	127	437	76	167
205	264	105	147	257	002	032

Multiplicaciones octales

La multiplicación se hace de la misma manera en que la realizaríamos con los decimales, solo que al multiplicar usamos la siguiente tabla y cuando se trata de sumar retomamos la tabla anterior de la suma octal para no caer en errores.

Tabla de multiplicación octal

*	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	10	12	14	16
3	0	3	6	11	14	17	22	25
4	0	4	10	14	20	24	30	34
5	0	5	12	17	24	31	36	43
6	0	6	14	22	30	36	44	52
7	0	7	16	25	34	43	52	61

Comencemos la explicación con un ejemplo:

	6	5		4	3	x	1	3	5
		4		1	3	5	7		
+	2	4		0	5	1			
	6	5		4	3				
1	1	5		6	3	6	7		

Explicación de la multiplicación

Primera línea de multiplicación

Comenzamos con los primeros números: 5 × 3 nos da 17, entonces ponemos el 7 y se acarrea 1; luego 5 × 4 es 24, pero se había acarreado 1 y se le suma, entonces se tiene 25, se pone el 5 y se acarrea el número 2; ahora sigue multiplicar 5 × 5, que es 31, más 2 de acarreo son 33,

se pone el 3 y se acarrea 3; toca multiplicar 5 × 6, que es igual a 36, más 3 de acarreo se suman con la tabla octal y nos da 41.

Segunda línea de multiplicación

En la segunda línea se multiplican 3 × 3, lo que nos da 11, se pone 1 y se sube 1 de acarreo; luego sigue multiplicar 3 × 4, que si buscamos en la fila 3 y columna 4 nos dará 14, más 1 que llevábamos de acarreo son 15, se pone el 5 y se sube el 1 de acarreo; después multiplicamos 3 × 5, que son 17, más 1 de acarreo tenemos 20, se pone el 0 y se sube 2 de acarreo, sigue multiplicar 3 × 6 y nos da 22, más 2 de acarreo nos da 24, y ya se pone el 24. La segunda línea queda así: 24051.

Tercera línea

Se multiplica por el 1, y como todo número multiplicado por 1 es el mismo número, se pone el número 6543 tal como está. Ahora se procede a la suma de los números, como se vio en el tema de las sumas.

Divisiones octales

Como en las multiplicaciones, el proceso de las divisiones es igual que en decimal, lo que cambia son las tablas de multiplicar y se resta en octal.

Ejemplos

```
        211
   ---------
351 | 765 45
      722
   ---------
      043 4
       35 1
   ---------
      00635
        351
   ---------
        264
   ---------
```

Operaciones con números hexadecimales

Al igual que los octales, para realizar todas sus operaciones se utilizan las siguientes tablas y se suman renglón por columna.

+	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
2	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11
3	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
4	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13
5	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14
6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
7	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
8	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17
9	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18
A	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19
B	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A
C	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

Suma hexadecimal

Como en el sistema hexadecimal, también se hace por medio de la tabla correspondiente de la suma para no confundir con la suma decimal.

Ejemplos

Ejemplo 1. En este ejemplo se suman E + E, si vemos en la tabla de suma hexadecimal, fila E con columna E es 1C, entonces se pone C y se acarrea 1; luego B + F es fila B con columna F, que da 1A y le sumamos 1 de acarreo, por lo que es 1B, se pone la B y se acarrea 1; luego E + A resulta 18, más 1 de acarreo da 19, se pone el 9 y se acarrea el 1; finalmente, B + C nos da 17 más 1 de acarreo es 18.

B E B E
C A F E
---
1 8 9 B C

Restas con hexadecimales

Primero debemos verificar que el número de arriba sea mayor que el de abajo, si no lo es intercambiamos posiciones. Como se puede observar en la tabla siguiente, en la primera columna la palabra BEBE es menor que CAFÉ, porque el primer dígito que está en el primer

número, una B, es menor que C. Entonces intercambiamos posiciones, luego procedemos a restar los números: en el ejemplo tenemos E − E, y nos da cero; en seguida está F − B, por lo que buscamos una F en la tabla de la suma, en el renglón B, y la encontramos en la columna 4, lo cual significa que B + 4 es F, por lo que ponemos el 4; después toca restar A − E, y observamos que A es menor que E, por lo tanto le pide prestado un número a C y se convierte en 1A, buscamos en la columna E el valor 1A y lo encontramos en la columna C, de modo que ponemos la C; por último, sigue restar B – B, recordemos que la C prestó uno y se quedó solo con B, por lo tanto B − B es igual a 0.

B E B E	C A F E
C A F E	B E B E
	0 C 4 0

Multiplicaciones hexadecimales

Para realizar las multiplicaciones se necesita la siguiente tabla.

Multiplicación

*	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	2	4	6	8	A	C	E	10	12	14	16	18	1A	1C	1E
3	3	6	9	C	F	12	15	18	1B	1E	21	24	27	2A	2D
4	4	8	C	10	14	18	1C	20	24	28	2C	30	34	38	3C
5	5	A	F	14	19	1E	23	28	2D	32	37	3C	41	46	4B
6	6	C	12	18	1E	24	2A	30	36	3C	42	48	4E	54	5A
7	7	E	15	1C	23	2A	31	38	3F	46	4D	54	5B	62	69
8	8	10	18	20	28	30	38	40	48	50	58	60	68	70	78
9	9	12	1B	24	2D	36	3F	48	51	5A	63	6C	75	7E	87
A	A	14	1E	28	32	3C	46	50	5A	64	6E	78	82	8C	96
B	B	16	21	2C	37	42	4D	58	63	6E	79	84	8F	9A	A5
C	C	18	24	30	3C	48	54	60	6C	78	84	90	9C	A8	B4
D	D	1A	27	34	41	4E	5B	68	75	82	8F	9C	A9	B6	C3
E	E	1C	2A	38	46	54	62	70	7E	8C	9A	A8	B6	C4	D2
F	F	1E	2D	3C	4B	5A	69	78	87	96	A5	B4	C3	D2	E1

La forma de realizar estas multiplicaciones es la siguiente: se hace la multiplicación como si fuera en decimal, pero en lugar de poner las tablas de los números decimales se toman las tablas de los hexadecimales. Cuando se hacen las sumas se utiliza también la tabla de suma, como se ve en la columna donde se suma el 4 con el 6 que nos da A.

1	F	3	2	x	B	0	E
			1	B	4	B	C
+			0	0	0	0	
	1	5	7	2	6		
	1	5	8	D	A	B	C

Divisiones hexadecimales

Para las divisiones lo primero que se hace es observar cuántos dígitos tiene el divisor, y se tantea cuántas veces cabe el divisor en los mismos dígitos del dividendo; se pone el número en el cociente y se multiplica por el divisor; luego se realiza la resta, y si la resta es más pequeña que el divisor estamos bien, si no se borra y se eleva el cociente un número más. Por supuesto, se usan las tablas de multiplicar y sumar hexadecimales.

```
        1 3 1
    ┌──────
90  │ A B  C D
      9 0
    ──────
      1 B C
      1 B 0
    ──────
        0 C D
          9 0
    ──────
            3  D
```

Aplicaciones de los sistemas numéricos en la computación

Los sistemas numéricos son fundamentales en el ámbito de la computación, ya que permiten representar y manipular datos en formatos entendibles para las máquinas. A continuación, se explica cómo se aplican los sistemas numéricos en la computación moderna.

Sistema binario (base 2) y representación de datos

En la computación, el sistema binario es crucial porque los dispositivos digitales operan a través de circuitos que pueden encontrarse en un estado de encendido o apagado, representado como 1 y 0. Estos valores binarios constituyen la base de los datos en cualquier computadora, ya que toda la información se procesa y almacena en bits, los cuales representan dígitos en binario (Tanenbaum y Austin, 2013). Además, el sistema binario permite representar instrucciones y datos, desde números hasta caracteres y gráficos, en un lenguaje que los circuitos electrónicos pueden entender (Stallings, 2017).

Sistema octal y su rol en codificación

Aunque su uso es menos común hoy en día, el sistema octal (base 8) fue usado en las primeras generaciones de computadoras para simplificar el trabajo con números binarios largos. Cada dígito octal representa tres bits binarios, lo que facilitaba el trabajo en sistemas de bajo nivel como el ensamblador y en la programación de *hardware* (Mano y Kime, 2013). En el caso de algunos sistemas operativos antiguos, como UNIX, el octal también se utiliza para representar permisos de archivos, proporcionando una notación compacta y accesible para operaciones de bajo nivel (Tanenbaum y Bos, 2014).

Sistema hexadecimal (base 16) y programación de bajo nivel

El sistema hexadecimal (base 16) es ampliamente utilizado en computación, especialmente en programación de bajo nivel y depuración, debido a su capacidad para representar grandes cantidades de datos en menos espacio. Dado que cada dígito hexadecimal equivale a cuatro bits, el sistema hexadecimal permite expresar secuencias binarias de una forma más compacta, lo cual es útil en la lectura y escritura de direcciones de memoria y en la representación de colores en la programación gráfica (Hennessy y Patterson, 2017). Así, los programadores y técnicos pueden trabajar eficientemente en la manipulación de datos binarios largos.

Sistema decimal (base 10) y aplicaciones de alto nivel

Aunque la computadora internamente utiliza sistemas como el binario o hexadecimal, los resultados y datos suelen traducirse al sistema decimal para que los usuarios puedan interpretarlos fácilmente. Esto se debe a que el sistema decimal es el más natural para los

humanos, y las interfaces de usuario suelen utilizarlo para mostrar datos financieros, cálculos científicos y otro tipo de información en un formato comprensible (Stallings, 2017). En este sentido, las aplicaciones de *software* suelen convertir las operaciones realizadas en binario o hexadecimal a una representación decimal para facilitar la interacción humana.

En resumen, los sistemas numéricos, especialmente el binario, octal, hexadecimal y decimal, son indispensables en la computación, ya que permiten la representación y manipulación eficiente de datos y código. Esta estructura numérica es la base del funcionamiento de las computadoras modernas, al facilitar tanto la comunicación entre máquinas como la interacción con el usuario.

Referencias

Andonegui Zabala, M. (2007). *El sistema numérico decimal*. Federación Internacional Fe y Alegría.

Clagett, M. (1989). *Ancient egyptian science: a source book*. University of Pennsylvania Press.

Closs, M. P. (1986). *Native American Mathematics*. University of Texas Press.

Coronado Padilla, J. A. (2014). *Sistemas numéricos residuales: fundamentos lógico-matemáticos* (1.ª ed.). Ediciones Unisalle.

Gardner, M. (1983). *Logic machines and diagrams*. Harvard University Press.

Hennessy, J. L., y Patterson, D. A. (2017). *Computer architecture: a quantitative approach* (6.ª ed.). Morgan Kaufmann.

Ifrah, G. (2000). *The universal history of numbers: from prehistory to the invention of the computer*. John Wiley y Sons.

Johnsonbaugh, R. (1999). *Matemáticas discretas* (4.ª ed.). Editorial Pearson.

Joseph, G. G. (2011). *The crest of the peacock: non-european roots of mathematics*. Princeton University Press.

Katz, V. J. (2009). *A history of mathematics: an introduction* (3.ª ed.). Addison-Wesley.

Knuth, D. E. (1997). *The art of computer programming: volume 2: seminumerical algorithms* (3.ª ed.). Addison-Wesley.

Mano, M. M., y Kime, C. R. (2013). *Logic and computer design fundamentals*. Pearson.

Menninger, K. (1992). *Number words and number symbols: a cultural history of numbers*. Dover Publications.

Robledo, I., Mendoza, A., y Perea, R. (s.f.). *Sistemas numéricos*. Universidad Autónoma de Ciudad Juárez.

Robson, E. (2008). *Mathematics in ancient Iraq: a social history*. Princeton University Press.

Rosen, K. H. (2012). *Matemática discreta y sus aplicaciones* (7ª ed.). McGraw-Hill.

Stallings, W. (2017). *Computer organization and architecture: designing for performance*. Pearson.

Tanenbaum, A. S. (2016). *Structured computer organization* (6.ª ed.). Pearson.

Tanenbaum, A. S., y Austin, T. (2013). *Structured computer organization*. Pearson.

Tanenbaum, A. S., y Bos, H. (2014). *Modern operating systems*. Pearson.

UNAD. (s.f.). *Sistemas numéricos*.