

CAPÍTULO II: REPRESENTACIÓN DE LA INFORMACIÓN

Tal vez el concepto más primitivo relacionado con la informática es el de número y relacionado con él la habilidad de contar. Para los hombres primitivos era relativamente fácil distinguir entre 2 y 5 árboles, pero mucho más difícil relacionar 4 árboles con 4 pájaros; es decir, separar los números de las cosas concretas que se cuentan o, en otras palabras, abstraer el concepto de número. Prueba de esta dificultad es la existencia de culturas que utilizaban distintos lenguajes numéricos dependiendo de los objetos que representaban.

Se tiene conocimiento de que el hombre, a través de distintas culturas, ha utilizado distintas bases de numeración, entre las que se encuentran las bases 4, 5, 10, 13, 18 y 20. Los pueblos indo-europeos también utilizaron el sistema de base 20, siendo una reminiscencia de este hecho la forma de escribir ciertos números franceses: 80 es quatre-vingts (cuatro veintes), 91 es quatre-vingt-onze (cuatro veintes once), etc...

Los sistemas de numeración, desde un punto de vista histórico, pueden clasificarse en sistemas aditivos y sistemas posicionales. Entre los primeros cabe destacar por su difusión, el egipcio y el romano. El sistema indo-arábigo (utilizado en la actualidad) es de tipo posicional. Los primeros documentos egipcios referentes a su notación numérica y métodos aritméticos datan del año 3.000 a.C., y eran de tipo jeroglífico.

El sistema primitivo romano, utilizaba 7 símbolos: M, D, C, L, X, V y I, y con ellos podía expresarse cualquier número menor de 5.000 con una secuencia de símbolos, sin repetir cada uno de ellos más de cuatro veces. En aquella época no se utilizaban las formas reducidas IV = IIII (por 4) y IX = VIIII (por 9). El número 3.524 se representaba así: MMMDXXIIII, y el orden de los símbolos no importaba, aunque se solían ordenar de mayor a menor. La operación de suma es inmediata (de ahí el nombre de *sistemas aditivos*), ya que consiste en mezclar los símbolos de los sumandos, ordenarlos de mayor a menor y simplificar (IIIIII = V, etc...). El producto resulta también muy sencillo, ya que se efectúa sumando los números que se obtienen haciendo el producto del multiplicando por cada símbolo individual del multiplicador.

Por el siglo I o II, los hindúes (en el este de Indochina) dieron en lo que respecta a los sistemas de numeración tres pasos trascendentales:

- **La representación posicional de las cifras.**
- **La base decimal.**
- **El concepto de cero como un dígito más.** Algunos científicos consideran que la invención del cero fue uno de los grandes logros de la humanidad. Se tiene constancia de que ya en el año 2.000 a.C. los babilonios utilizaban un símbolo (predecesor del cero) para las posiciones vacías dentro del número, pero les era imposible utilizar el cero aisladamente en las posiciones finales (por ejemplo, para representar las dos últimas posiciones de 100).

Esta conjunción de ideas dio al mundo una notación flexible y eficiente. Con ella todo número, por grande que sea, puede representarse por una secuencia ordenada de símbolos tomados de un conjunto de diez.

Las primeras constancias escritas de la introducción del sistema indo-arábigo en Europa se encuentran en España en el texto Albelda Cloister (976), y las cifras numerales también aparecen en otro escrito español de 992. Realmente quien difundió el sistema de numeración en Occidente fue el gran matemático Leonardo de Pisa (más conocido por el nombre de Fibonacci) hacia el 1.200, quien hizo aportaciones de gran interés en la búsqueda de algoritmos para la resolución de problemas concretos.

El vocablo **algoritmo** se deriva del nombre de su creador "Al-Khowârizmî", matemático árabe.

Un paso de gran relevancia en los métodos de cálculo lo dio hacia 1.580 el francés François Viète, que propuso la utilización de las letras para representar números desconocidos, lo que supuso un paso trascendental en el desarrollo del álgebra.

Históricamente la siguiente invención de gran importancia se debe a John Napier, que desarrolló la teoría de logaritmos, como ayuda al cálculo, ya que con ellos la multiplicación se podía transformar en sumas, la potenciación en productos, etc....

Paralelamente al desarrollo de los sistemas de representación de los números y de los métodos de cálculo, se fueron ideando formas de sistematización del razonamiento. Los griegos (siglo IV a.C.) sentaron las bases de la **lógica formal**. Ellos desarrollaron la geometría como un sistema de deducción lógica (partiendo de axiomas, postulados, teoremas, etc...) y describieron una metodología para la realización de razonamientos correctos (silogismos, etc...).

Los babilonios y egipcios utilizaron otra metodología distinta a la lógica, la metodología experimental, con la que a base de procesos iterativos de prueba-error obtuvieron tablas de multiplicar, tablas de raíces, tablas de potencias, tablas contables (interés compuesto), etc...

El siguiente avance que merece destacarse en esta reseña histórica se debe a Leibniz (1646-1716), que sentó las bases de la **lógica simbólica** y sugirió la utilización de la aritmética binaria para hacer automáticamente razonamientos. Georges Boole (1815-1869) publicó dos libros de gran importancia ("Análisis matemático de la lógica, como ensayo hacia un cálculo de razonamiento deductivo" e "Investigación sobre las leyes del pensamiento"). A partir de profundizar en el pensamiento de Aristóteles, y de sus amplios conocimientos matemáticos, asociaba a funciones lógicas (unión, intersección, complementación, etc....) y a variables lógicas (verdadero/falso), símbolos matemáticos (+, *, -, 1, 0). Desarrolló un álgebra (Álgebra de Boole) aplicable a este tipo de funciones y variables. Sus trabajos fueron fundamentales para el desarrollo de la lógica matemática (en cierta medida un álgebra para la lógica) y de procedimientos sistemáticos para el diseño de los circuitos que componen un computador.

El ser humano siempre ha necesitado medios para hacer cálculos y procesar la información. La complejidad de estos se ha ido acrecentando en el transcurso del tiempo, conforme surgían nuevas necesidades, y ha estado subordinada a los progresos de la tecnología. En cuanto al cálculo, primero surgieron los instrumentos aritméticos, como el ábaco, a partir de los que se ha llegado a las calculadoras y ordenadores actuales.

El origen del procesamiento automático de la información, se remonta al año 1896 cuando Herman **Hollerith** (1860-1929) fundó una empresa que daría lugar a **IBM**. Actualmente casi todas las necesidades de cálculo, se han visto satisfechas con los ordenadores.

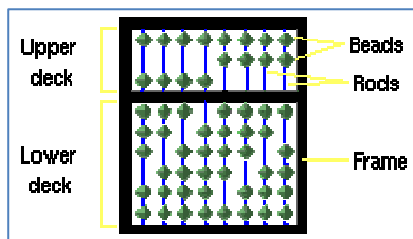


Es difícil determinar el punto de inicio para una síntesis histórica de la informática, por cuanto son muchos los trabajos y descubrimientos que trajeron como consecuencia la construcción del primer ordenador. Desde tiempo inmemorial los hombres se han valido de instrumentos para realizar cálculos y para almacenar y procesar información. La primera herramienta que servía para contar y al mismo tiempo para representar las cantidades contadas fueron los dedos, dando origen al sistema decimal de

numeración.

El hombre primitivo usó piedrecillas para representar números y hacer sumas sencillas. 500 años a.n.e., apareció el ábaco inventado y reinventado por culturas distintas en el espacio y en el tiempo, como los aztecas y los sumerios. El ábaco ruso es decimal, dispone de diez anillos de madera en cada columna. En el chino el tablero está dividido en dos zonas, "cielo" y "tierra", con dos y cinco bolas respectivamente. El japonés consta de cuatro y una bola respectivamente. En Japón existe un Instituto de Investigación del ábaco y un Comité Central de Operadores de ábacos. El 12 de noviembre de 1946 compitieron, el soldado Wood del ejército de EE.UU., que era el operador de máquinas de calcular más experto, con una calculadora eléctrica de las de su época y Kiyoshi Matsuzaki del Departamento de Ahorros del Ministerio de Administración Postal, dotado de un ábaco. Se trataba de resolver cinco cálculos comprendiendo las cuatro operaciones elementales, la victoria fue para el japonés, por cuatro frente a uno.

EL ÁBACO



Considerado como el instrumento más antiguo de cálculo, adaptado y apreciado en diversas culturas. El origen del ábaco está literalmente perdido en el tiempo. En épocas muy tempranas el hombre primitivo encontró materiales para idear instrumentos de conteo. Es probable que su inicio fuera una superficie plana y piedras que se movían sobre líneas dibujadas con polvo. Hoy en día se tiende a pensar que el origen del ábaco se encuentra en China, donde el uso de este instrumento aun es notable al igual que en Japón.

La palabra ábaco es latina y tiene sus orígenes del griego "abax" o "abakon", que significa "superficie plana" o "tabla", es posible que se haya originado de la palabra Semítica Abaq que significa "polvo". Otros nombres son: del ábaco Chino es "Suan Pan", el Japonés es "Soroban", en Corea "Tschu Pan", en Vietnam "Ban Tuan" o "Ban Tien", en Rusia "Schoty", Turquía "Coulba" y Armenia "Choreb".

Debido a que gran parte de la aritmética se realizaba en el ábaco, el término ábaco ha pasado a ser sinónimo de aritmética, y encontramos tal denominación en Leonardo de Pisa Fibbonacci (1170-1250) en su libro "Liber Abaci" publicado en 1202, que trata del uso de los números indo-arábigos.

Muchas culturas han usado el ábaco o el tablero de conteo, aunque en las culturas europeas desapareció al disponerse de otros métodos para hacer cálculos, hasta tal punto que fue imposible encontrar rastro de su técnica de uso. Las evidencias del uso del ábaco son comentarios de los antiguos escritores griegos. Por ejemplo, Demóstenes (384-322) escribió la necesidad del uso de piedras para realizar cálculos difíciles de realizar en la cabeza. Y los métodos de cálculo encontrados en los comentarios de Herodoto (484-425), hablando de los egipcios decía: "Los Egipcios mueven su mano de derecha a izquierda en los cálculos, mientras los griegos lo hacen de izquierda a derecha".

Algunas de las evidencias físicas de la existencia del ábaco se encontraron en épocas antiguas de los griegos por las excavaciones arqueológicas. En 1851, se encontró una gran ánfora de 120 cm. de alto, se la denominó como "Vaso de Darío" y entre los dibujos tiene una figura que representa un contador que realiza los cálculos. La segunda muestra arqueológica es un auténtico tablero de conteo encontrado en 1846 en la isla de Salamis, el tablero de Salamis probablemente usado en Babilonia 300 a.n.e., es una gran pieza de mármol de 149 cm. de largo por 75 cm. de ancho, con inscripciones que se refieren a ciertos tipos de monedas de la época, este tablero está dividido en dos partes.

Se sabe que los romanos empleaban su ábaco con piedra caliza o mármol, para las cuentas a las que denominaron "calculi". Esta palabra es la raíz de la palabra cálculo".

En el siglo XIII se estandarizó una mesa de ábaco en Europa, consistiendo en una mesa cubierta de paño en la que se dibujaban unas líneas con tiza o tinta. Existieron dos intentos por reemplazar la mesa de ábaco por otros más modernos. El primero fue ideado por el filósofo romano Boethius, quien escribió un libro sobre geometría dedicando un capítulo al uso del ábaco, describió como en lugar de emplear cuentas se podía representar el número con sólo una cuenta que tuviese los dígitos del 1 al 9 marcados. El segundo intento fue realizado por el monje Gerbert de Avrillac (945-1003), quien fue papa con el nombre de Silvestre II. Tomó ideas del libro de Boethius, y describió el uso de una nueva forma de ábaco en el año 1000. Ninguno de estos dos ábacos fue popular.

La mesa de ábaco fue usada extensamente en Bretaña, al igual esta fue abandonada por la mayoría de la gente. El libro "The Ground of Artes" escrito por Robert Recorde (1510-1558) en 1542, claramente muestra el método de aritmética con la mesa de ábaco.

Conforme los numerales indo-arábigos aparecieron en Europa el uso de la mesa de ábaco desapareció por completo, cuando los soldados de Napoleón invadieron Rusia en 1812, trajeron ábacos como trofeos o recuerdos del país.

En otras partes del mundo, se encuentra en China la primera evidencia del inicio del ábaco chino que se descubrió, fueron cuentas de cerámica hechas en el occidente de la dinastía Zhou¹ con más de 3000 años. Respecto a los materiales históricos a mano, el libro que registra el comienzo del cálculo con un ábaco se llama Crónica Aritmética escrito por Xu Yue en el oriente de la dinastía Han (206-220), hace 2000 años. Esto indica que el ábaco tenía una cuenta en la parte superior y cuatro en la inferior.

Los ábacos modernos existieron en la dinastía Song (960-1279) lo que puede ser verificado por alguna evidencia, por ejemplo, en una pintura de Wang Xhenpeng's, muestra el uso extenso entre la gente del sur de la dinastía Song.

Durante la dinastía mongol Yuan (1279-1368) los ábacos tuvieron una etapa donde se fueron popularizando paulatinamente en todo el país, posteriormente entró en la etapa en la que su uso ya era algo común a mediados de la dinastía Ming (1368-1644) y la técnica de uso paso a ser un sistema algorítmico completo. Un libro escrito por Wu Ching-Hsin-Min en 1450, tiene descripciones acerca del ábaco, así como una gran número de libros publicados a finales de la dinastía Ming, que aseguran el hecho que el ábaco entró en el uso popular. Existen dos trabajos representativos en el cálculo del ábaco en la Dinastía Ming. Uno fue Wang Wensu's Principios matemáticos 1524, y el otro es Cheng Dawei's reglas generales del método de conteo 1592, los

cuales plantearon un mayor papel en extender el uso del ábaco. Durante el período de la dinastía Ming, el ábaco chino se propagó hacia Corea en el 1400 y en Japón en el 1600, así como al sureste de Asia.

Durante la dinastía Ming había un solo tipo de ábaco en China, con una cuenta en la parte superior y cinco en la parte inferior, fue encontrado en la tumba de Lu Weizhen (1543-1610). Después de la dinastía Qing (1644-1912), el ábaco contó con dos cuentas en la parte superior y cinco en la parte inferior, fue extensamente usado como actualmente ha sido, mientras que el ábaco japonés se diseñó empleando una cuenta en la parte superior (cielo) y cuatro en la inferior (tierra).

A finales de la edad media los mongoles propagaron el uso del ábaco en Rusia, que provenía de los chinos y los tártaros.

Un hecho muy importante del uso y la potencia del ábaco fue que el 12 de Noviembre de 1946, una competencia, entre el japonés Kiyoshi Matsuzaki del Ministerio Japonés de comunicaciones utilizando un ábaco japonés y el americano Thomas Nathan Wood de la armada de ocupación de los EE.UU. con una calculadora electromecánica, fue llevada a cabo en Tokyo, bajo patrocinio del periódico del ejército americano (U.S. Army), Stars and Stripes. Matsuzaki utilizando el ábaco japonés resultó vencedor en cuatro de las cinco pruebas, perdiendo en la prueba con operaciones de multiplicación.

El 13 de Noviembre de 1996, los científicos Maria Teresa Cuberes, James K. Gimzewski, y Reto R. Schlittler del laboratorio de IBM de Suiza de la división de investigación, construyeron un ábaco que utiliza como cuentas moléculas cuyo tamaño es inferior a la millonésima parte del milímetro. El "dedo" que mueve las cuentas moleculares es similar a una aguja cónica que en su extremo más puntiagudo alberga un átomo.

Las fechas de inicio de la dinastía Zhou son dudosas, y varían entre los años 1122, 1050 y 1027 a.n.e., cuando al parecer expulsaron a la débil dinastía Shang. En cuanto a la fecha de su fin, casi todo el mundo coincide en señalar el año 221 a.n.e.

Antes de aparecer las calculadoras surgieron otros dispositivos de entre los que cabe comentar dos, en los que el matemático escocés [John Neper](#) (1550-1617) tuvo un papel destacado. Es conocido por la invención de los logaritmos en 1614, que dieron origen a la regla de cálculo, cuya paternidad es tema de controversia, no obstante el primero en usarla, en 1621, fue el sacerdote inglés [William Oughtred](#) (1575-1660). En 1617 Neper dio a conocer un instrumento sencillito para realizar multiplicaciones basándose en sumas, llamado rodillos de Neper, idea que aparecía varios siglos antes en libros árabes.

La necesidad de calcular sin errores dio lugar a la calculadora, la mecánica es una especie de ábaco, pero con ruedas dentadas en lugar de varillas y bolas, dotada de

un mecanismo para el transporte de las unidades que se lleven, de una posición digital a la siguiente más significativa.



Hasta hace pocas décadas se creía que el filósofo francés [Blas Pascal](#) (1623-1662) había sido el creador de la calculadora. Pascal diseñó su "machina arithmetica", posteriormente denominada Pascalina, a la edad de 19 años, para que su padre que era recaudador de impuestos tuviera tiempo libre para jugar con él a la paume.

[Leonardo Da Vinci](#) (1452-1519) diseñó una sumadora que fue reconstruida en 1967 a partir de uno de sus códices. En 1935 el historiador Franz Hammer, revisando la correspondencia del astrónomo [Johannes Kepler](#) descubrió que el alemán [Willebrord Snellius](#) (1592-1635) había inventado una calculadora que era una combinación de los rodillos de Neper con una sumadora-restadora similar a la de Pascal, obviamente no sólo era superior a la pascalina, sino que se construyó el año en que nació Pascal.

El primero en construir una calculadora, en 1671, fue el filósofo y matemático alemán [Gottfried Leibniz](#) (1646-1716), inventor junto con [Isaac Newton](#) del cálculo infinitesimal, aunque de forma independiente. Fue denominada calculadora universal, su elemento característico era un tambor cilíndrico con nueve dientes de longitud variable, llamado rueda escalonada, que se encuentra en prácticamente todas las calculadoras mecánicas posteriores, incluso las del siglo XX. Las técnicas de producción tan poco eficientes de aquella época, impidieron que el invento de Leibniz se fabricara masivamente. Se llegaron a construir 1500 unidades, pero hubo que esperar hasta 1820 para que Carlos Thomas, director de una aseguradora diseñara un modelo capaz de ser producido a bajo coste y a escala industrial.



En 1872 el estadounidense Frank Baldwin construyó una calculadora a la que años más tarde le añadió la denominada rueda Odhner. Esta fue la antecesora de la clásica calculadora de sobremesa, con manecilla lateral, difundida universalmente desde 1910 y que todavía se encuentra en rastros. De ella deriva la popular caja registradora inventada en 1879 por [James Ritty](#), comercializada bajo la marca National y una sumadora provista de impresora inventada por [William Borroughs](#) (1855-1898) en 1884, fundador de la empresa que llevó su apellido.

En 1878 el periodista y escritor gallego, afincado en EE.UU., [Ramón Verea García](#) (1833-1899) patentó en Nueva York una calculadora por la que se le otorgó la medalla de oro de la exposición de Matanzas (Cuba). Aseguraba que no había fabricado la máquina para patentarla y venderla, sino para demostrar que era posible que un español pudiera inventar tan bien como un norteamericano. A partir de entonces sólo

se dedicó al periodismo, combatiendo la política de colonialismo de EE.UU., por lo que tuvo que exiliarse en Guatemala y posteriormente en Argentina.

Cálculo Matemático.

Una calculadora no es un dispositivo automático, lo cual implica que requiere la acción constante de un operador, que es un obstáculo para la velocidad y fiabilidad de los resultados.



En 1812 el matemático inglés [Charles Babbage](#) (1792-1871), habiendo constatado que las tablas trigonométricas estaban plagadas de errores al haber sido calculadas a mano, concibió la denominada máquina de diferencias, un instrumento mecánico para calcular e imprimir tablas de funciones. En realidad se trataba de una máquina que calculaba el valor numérico de una función polinómica sobre una progresión aritmética, pues las funciones se pueden aproximar por polinomios.

Tras una serie de fracasos, en 1832 Babbage desarrolló el proyecto de la máquina analítica. Se trataba de un ordenador mecánico de propósito general, preparado para realizar cualquier tipo de cálculo mediante un programa adecuado. Sus elementos fundamentales serían: una memoria para 1000 números de 50 cifras, una unidad aritmético lógica para los cálculos, una unidad de control para que las operaciones se realizasen en el orden correcto, lectoras de fichas perforadas (que ya se usaban desde hace un siglo en los telares) para la entrada de datos y una impresora para la salida de resultados.

Una amiga y colaboradora, la señorita [Ada Augusta Byron](#) (1815-1852), condesa de Lovelace, publicó una serie de programas para resolver ecuaciones trascendentes e integrales definidas, con la máquina analítica. En dichos programas se hacía uso de bifurcaciones, hacia delante y hacia atrás y de bucles. Fue la primera programadora de la historia, por eso el departamento de Defensa de EE.UU. puso su nombre al lenguaje de programación de uso obligatorio en sus dependencias.



Es sorprendente que a alguien se le ocurriera diseñar un ordenador hace más de un siglo y medio. Aunque nunca se llegó a construir esta máquina por falta de precisión en algunas piezas. Babbage tenía manía a los organilleros, y al morir los periódicos londinenses destacaron ese detalle.

Entre sus sucesores destaca el ingeniero santanderino **Leonardo Torres Quevedo** (1852-1936). Logró renombre universal gracias a sus inventos. Construyó transbordadores (uno en las cataratas del Niágara), un aparato teledirigido por ondas de radio, un globo dirigido y semirrígido, usado por franceses e ingleses durante la Primera Guerra Mundial y un sinfín de máquinas para cálculo científico. De estos destacan los **aritmómetros** en los que introdujo la aritmética de punto flotante, eran máquinas de cálculo matemático sobre la base de relés, y dotadas de memoria, que se gobernaban a distancia mediante una máquina de escribir, la cual servía para entrar operandos, operaciones y para obtener los resultados. Asimismo realizó estudios sobre los hoy denominados robots, y sus aplicaciones en la industria, por lo cual no sólo es considerado un precursor de la informática sino también de la cibernética; como ejemplo práctico construyó una máquina de jugar al ajedrez, un autómatas capaz de dar mate de rey y torre contra rey y que reaccionaba ante las jugadas irreglamentarias del contrario.

En los años 1920 tuvo en sus manos el dar a España la primacía en la informática, si no sucedió fue porque en aquella época no hacía falta. La necesidad de un ordenador surgió con la Segunda Guerra Mundial, por lo que se construyeron los primeros ordenadores basándose en los trabajos de Babbage y de Torres Quevedo.

Una innovación muy importante, y en principio ajena a la Informática, tuvo lugar a principios del siglo XIX, supuso la **tarjeta perforada**, debida a Joseph Jacquard en 1.801.

En 1.822 el inglés Charles Babbage diseñó una **máquina de diferencias** para producir tablas de navegación. Esta máquina puede considerarse que era un computador digital con un programa fijo. A Babbage, mientras diseñaba la máquina de diferencias, se le ocurrió la idea de que podía modificar las interconexiones entre los registros durante el proceso de cómputo, con lo que obtendría un computador de uso general. De Jacquard obtendría la **tarjeta perforada** para controlar interconexiones y dar la entrada de datos. Los elementos que realizaban las operaciones aritméticas los denominó **taller** ("mill" en inglés), necesitando otros elementos para almacenar los números, a los que denominó **almacén** ("store" en inglés). Puede constatar, que se adelantó a la concepción actual de un computador en cuanto a que proponía cuatro elementos fundamentales: *entradas, salidas, unidad aritmético-lógica (el taller) y memoria (el almacén)*, y en cuanto que su máquina estaba concebida como un computador universal completamente automático y capaz de resolver gran cantidad de problemas, no para uno concreto. El sistema ideado por Babbage (1.833) se denominó **máquina analítica**, y no llegó a construirla, ya que era de una complejidad mecánica y se presentaban situaciones en las que se necesitaba el movimiento simultáneo de una gran cantidad de engranajes, y la mecánica de precisión de su época no estaba lo suficientemente desarrollada.

Un problema grave que se le presentó al Gobierno de Estados Unidos al final de la década de los años 1.880 era que con los medios que disponía preveían que tardarían en realizar el censo de 1.880 doce años, con lo que se solaparía con el de 1.890.

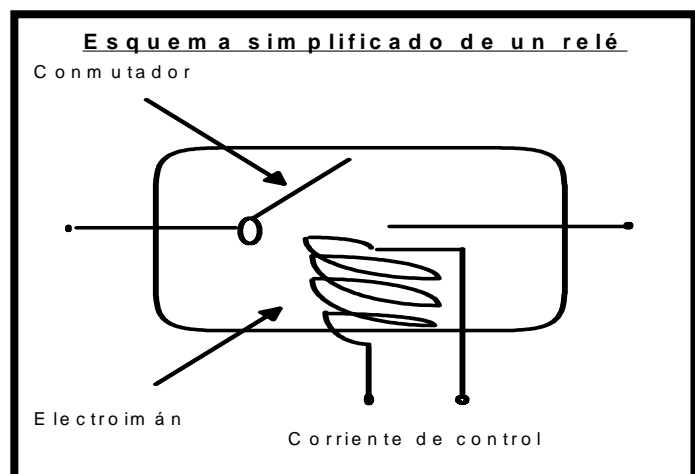
Hermann Hollerith, que trabajaba en la oficina de censos, concibió la idea de usar tarjetas perforadas para mecanizar el procesamiento de datos de los censos, desarrollando así una máquina llamada **tabuladora**, en la que las tarjetas van posicionándose una a una en una estación de lectura, cuyo soporte es una placa conductora. Unas varillas metálicas telescópicas entran en contacto con la superficie superior de la tarjeta; en los sitios que hay perforación las varillas tocan la placa metálica, cerrándose así un contacto eléctrico. Este circuito provoca el desplazamiento de un dial (existe un dial por cada posible perforación), que va contabilizando el número de tarjetas que tienen perforación en la posición correspondiente y hace que la tarjeta vaya al cajetín de clasificación correspondiente. Esta máquina tuvo un gran éxito, exponiéndose en la exposición universal de París de 1.889, e hizo posible que el censo de 1.890 se efectuase en tres años.

A partir de entonces se incrementa el desarrollo de distintos tipos de calculadoras mecánicas y electromecánicas basadas en las ideas de Pascal, Leibniz y Hollerith. Se crean varias empresas que diseñan y comercializan los nuevos productos: IBM en 1.924, Remington-Ran en 1.927, Bull (en Francia) en 1.931, etc...

El profesor Howard Aiken de la Universidad de Harvard construyó, entre 1.937 y 1.944, un computador denominado Mark I, que fue el primer calculador digital automático de uso general. Estaba principalmente proyectado para realizar tablas matemáticas y realizaba sumas y restas de 23 dígitos en 3 décimas de segundo, multiplicaciones en seis segundos y divisiones en doce. Los elementos principales de este computador eran relés.

Los **relés** son conmutadores que contienen un contacto mecánico que puede abrir o cerrar un circuito eléctrico, aplicando una pequeña corriente en un electroimán (que atrae o repele la lengüeta de contacto).

El Mark I utilizaba un gran número de tabuladoras Hollerith y una lectora de cinta de papel, en la que se perforaban las instrucciones, para controlar su funcionamiento.



Por esa época Claude E. Shannon presentó en el Instituto Tecnológico de Massachusetts (MIT) su tesis doctoral y publicó un trabajo (1.938) en el que se proponía aplicar el álgebra de Boole binaria para el diseño sistemático de circuitos lógicos complejos utilizando conmutadores. De esta forma, con conmutadores (relés, en aquella época) y aplicando los procedimientos ideados por Shannon, se pueden implementar de forma sistemática las funciones aritméticas, lógicas y de control que necesita un computador.

Datos.

Los "datos" son conjuntos de símbolos utilizados para expresar o representar un valor numérico, un hecho, un objeto o una idea; en la forma adecuada para ser objeto de tratamiento. En Informática el concepto de dato es mucho más amplio que en cualquier otra ciencia, ya que un dato puede ser cualquier tipo de información, bien sea simple o compuesta (un número, una magnitud, un título de un libro, una matrícula de coche, un nombre completo de persona, etc...). Los datos pueden ser captados directamente por el computador u ordenador (términos sinónimos) o pueden ser dados en forma de números y letras (grafismos).

Los grafismos (caracteres) resultan muy útiles, dada la gran variedad de información que con ellos se puede representar, siendo una de las formas más habituales de transmisión, comunicación o almacenamiento de información en nuestra sociedad actual: el lenguaje escrito. Los grafismos utilizados normalmente en Informática son los caracteres numéricos (las diez cifras decimales -0 a 9-), los caracteres alfabéticos (a-z) y los caracteres especiales (símbolos ortográficos, aritméticos y otros). Cualquier información (datos o instrucciones) puede expresarse utilizando caracteres, y así ser introducida en el ordenador. De igual forma, usualmente el ordenador nos dará los resultados en forma escrita, utilizando caracteres.

Con frecuencia los resultados o salidas de un programa, se denominan también datos, los cuales pueden ser utilizados posteriormente como datos de un programa posterior. Es decir, la palabra "dato" se utiliza como contraposición a "instrucción". El ordenador actúa con dos tipos de informaciones: "instrucciones" -que indican a la máquina que es lo que tiene que hacer- y "datos" -que son los elementos sobre los que actúa o genera el programa-.

La informática, como disciplina, utiliza los métodos y procedimientos de los desarrollos teóricos, experimentales y de diseños, por lo que es tanto una ciencia como una ingeniería. La disciplina de informática, es el cuerpo de conocimiento que trata del diseño, análisis, implementación, eficiencia y aplicación de procesos que transforman la información.

Codificación de la información.

En Informática es frecuente codificar la información. **Codificación** "es una transformación que representa los elementos de un conjunto mediante los de otro, de forma tal que a cada elemento del primer conjunto le corresponde un elemento distinto del segundo".

Ejemplos de códigos son: el código de países de matrículas de coches, el código de enfermedades asignado por la OMS, el número de DNI, el número de identificación de alumno, etc.

Con los códigos se puede comprimir y estructurar la información. La matrícula de un coche comprime la información del propietario, domicilio, color, marca, fecha de matrícula, fecha de compra, etc.

Pueden definirse códigos con "significado", así el código-matrícula de coche no se da al azar, sino que un/unos carácter/es corresponden al país, el número y las letras posteriores corresponden al orden temporal y secuencial en que se matriculó.

En el interior de los computadores la información se almacena y se transfiere de un sitio a otro según un código que utiliza dos valores (un código binario) representados por 0 y 1. En la entrada y salida del computador se efectúan automáticamente los cambios de código oportunos para que en su exterior la información sea directamente comprendida por los usuarios.

La unidad más elemental de información es un valor binario, conocido como Bit.

Bit es el acrónimo de *Binary digit*. (dígito binario). Un bit es un dígito del sistema de numeración binario. La Real Academia Española (RAE) ha aceptado la palabra bit con el plural bits.

Mientras que en el sistema de numeración decimal se usan diez dígitos, en el binario se usan sólo dos dígitos, el 0 y el 1. Un bit o dígito binario puede representar uno de esos dos valores, **0** ó **1**.

Podemos imaginarnos un bit como una bombilla que puede estar en uno de los siguientes dos estados:











Memoria de computadora de 1980 donde se pueden ver los bits físicos. Este conjunto de unos 4x4 cm. corresponden a 512 bytes.

El bit es la unidad mínima de información 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, masculino o femenino, amarillo o azul, etc. Basta con asignar uno de esos valores al estado de "apagado" (0), y el otro al estado de "encendido" (1).

Combinaciones de bits

Hay 4 combinaciones posibles con dos bits

| Bit 1 | Bit 0 |
|---|---|
|  0 |  0 |
|  0 |  1 |
|  1 |  0 |
|  1 |  1 |

Con un bit podemos representar solamente dos valores. Para representar o [codificar](#) más información en un dispositivo digital, necesitamos una mayor cantidad de bits. Si usamos dos bits, tendremos cuatro combinaciones posibles:

- **0 0** - Los dos están "apagados"
- **0 1** - El primero (de derecha a izquierda) está "encendido" y el segundo "apagado"
- **1 0** - El primero (de derecha a izquierda) está "apagado" y el segundo "encendido"
- **1 1** - Los dos están "encendidos"

Con estas cuatro combinaciones podemos representar hasta cuatro valores diferentes, como por ejemplo, los colores rojo, verde, azul y negro.

A través de secuencias de bits, se puede [codificar](#) cualquier valor [discreto](#) como números, palabras, e imágenes. Cuatro bits forman un [nibble](#), y pueden representar hasta $2^4 = 16$ valores diferentes; ocho bits forman un [octeto](#), y se pueden representar hasta $2^8 = 256$ valores diferentes. En general, con n número de bits pueden representarse hasta 2^n valores diferentes.

Nota: Un [byte](#) y un [octeto](#) no son la misma cosa. Mientras que un octeto siempre tiene 8 bits, un byte contiene *un número fijo de bits*, que no necesariamente son 8. En los computadores antiguos, el byte podría estar conformado por 6, 7, 8 ó 9 bits. Hoy en día, en la inmensa mayoría de los computadores, y en la mayoría de los campos, un byte tiene 8 bits, siendo equivalente al octeto, pero hay excepciones.

Valor de posición.

En cualquier sistema de numeración posicional, el valor de los dígitos depende del lugar en el que se encuentren.

En el sistema decimal, por ejemplo, el dígito 5 puede valer 5 si está en la posición de las unidades, pero vale 50 si está en la posición de las decenas, y 500 si está en la posición de las centenas. Generalizando, cada vez que nos movemos una posición hacia la izquierda el dígito vale 10 veces más, y cada vez que nos movemos una posición hacia la derecha, vale 10 veces menos. Esto también es aplicable a números con decimales.

| | | |
|---------|---------|---------|
| +-----+ | +-----+ | +-----+ |
| Centena | Decena | Unidad |
| +-----+ | +-----+ | +-----+ |
| x 100 | x 10 | x 1 |
| +-----+ | +-----+ | +-----+ |

Por tanto, el número 153 en realidad es: 1 centena + 5 decenas + 3 unidades, es decir,

$$100 + 50 + 3 = 153.$$

En el sistema binario es similar, excepto que cada vez que un dígito binario (bit) se desplaza una posición hacia la izquierda vale el doble (2 veces más), y cada vez que se mueve hacia la derecha, vale la mitad (2 veces menos).

| | |
|-----------------------|------------------|
| +---+---+---+---+---+ | Valor del bit |
| 16 8 4 2 1 | <-- de acuerdo a |
| +---+---+---+---+---+ | su posición |






Abajo vemos representado el número 19.

$$16 + 2 + 1 = 19.$$

| 16 | 8 | 4 | 2 | 1 | <-- Valor de posición |
|---|---|---|---|---|--|
|  |  |  |  |  | Representación gráfica de los bits como bombillas encendidas y apagadas |
| 1 | 0 | 0 | 1 | 1 | <-- Dígitos binarios (bits) |

También se pueden representar valores "decimales" (números reales, de punto flotante). Abajo vemos el número 5.25 representado en forma binaria.

$$4 + 1 + 0.25 = 5.25$$

| 4 | 2 | 1 | 1/2 | 1/4 | <-- Valor de posición |
|---|---|---|---|---|--|
|  |  |  |  |  | Representación gráfica de los bits como bombillas encendidas y apagadas |
| 1 | 0 | 1 | 0 | 1 | <-- Dígitos binarios (bits) |

Aunque la representación de números reales no es exactamente como lo que se muestra arriba, el esquema da una idea del concepto.

Subíndices

Cuando se trabaja con varios sistemas de numeración o cuando no está claro con cual se está trabajando, es típico usar un subíndice para indicar el sistema de numeración con el que se ha representado un número. El 10 es el subíndice para los números en el sistema decimal y el 2 para los del binario. En los ejemplos de arriba se muestran dos números en el sistema decimal y su equivalente en binario. Esta igualdad se representa de la siguiente manera:

- $19_{10} = 10011_2$
- $5.25_{10} = 101.01_2$

Bits más y menos significativos

Un conjunto de bits, como por ejemplo un [byte](#), representa un conjunto de elementos ordenados. Se llama [bit más significativo](#) (MSB) al bit que tiene un mayor peso (mayor valor) dentro del conjunto, análogamente, se llama [bit menos significativo](#) (LSB) al bit que tiene un menor peso dentro del conjunto.

En un Byte, el bit más significativo es el de la posición 7, y el menos significativo es el de la posición 0

| | |
|-----------------------------------|--|
| +---+---+---+---+---+---+---+---+ | |
| 7 6 5 4 3 2 1 0 | <-- Posición del bit |
| +---+---+---+---+---+---+---+---+ | |
| 128 64 32 16 8 4 2 1 | <-- Valor del bit de acuerdo a su posición |
| +---+---+---+---+---+---+---+---+ | |
| | |
| (+) | (-) |
| ----- | ----- |
| Bit más significativo | Bit menos significativo |

En una palabra de 16 bits, el bit más significativo es el de la posición 15 y el menos significativo el de la posición 0.

| | | | | | | | | | | | | | | | | |
|----------|----------|----------|----------|----------|----------|-----|-----|-----|----|----|----|---|---|---|---|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | <-- Posición del bit |
| 2^{15} | 2^{14} | 2^{13} | 2^{12} | 2^{11} | 2^{10} | 512 | 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 | <-- Valor del bit de acuerdo a su posición |
| | | | | | | | | | | | | | | | | --- |
| | | | | | | | | | | | | | | | | Bit menos significativo |
| | | | | | | | | | | | | | | | | Bit más significativo |

Tomemos, por ejemplo, el número decimal 27 codificado en forma binaria en un octeto:

$$27 = 16 + 8 + 2 + 1 = 2^4 + 2^3 + 2^1 + 2^0 \rightarrow 00011011$$

Aquí, el primer '0', el de la izquierda, (que se corresponde con el coeficiente de 2^7), es el bit más significativo, siendo el último '1', el de la derecha, (que se corresponde con el coeficiente de 2^0), el menos significativo.

En cualquier caso, el bit más significativo es el del extremo izquierdo y el menos significativo el del extremo derecho. Esto es análogo al sistema decimal, en donde el dígito más significativo es el de la izquierda y el menos significativo el de la derecha, como por ejemplo, en el número 179, el dígito más significativo, el que tiene mayor valor, es el 1, (el de las centenas), y el menos significativo, el 9, (el de las unidades).

Little endian y Big endian

En los computadores cada [byte](#) se identifica con su posición en la [memoria](#) (dirección). Cuando se manejan números de más de un byte, éstos también deben estar ordenados. Este aspecto es particularmente importante en la programación en código máquina, ya que algunas máquinas consideran el byte situado en la dirección más baja el menos significativo (arquitectura *little endian*, como los procesadores [Intel](#)) mientras que otras consideran que ése es el más significativo (arquitectura *big endian*, como los procesadores [Motorola](#)). De este modo, un byte con el número decimal 27 se almacenaría en una máquina *little endian* igual que en una máquina *big endian*, ya que sólo ocupa un byte. Sin embargo, para números más grandes los bytes que los representan se almacenarían en distinto orden en cada arquitectura.

Por ejemplo, consideremos el número hexadecimal entero AABBCDD, de 32 bits (4 bytes), localizado en la dirección 100 de la memoria. El número ocuparía las posiciones desde la 100 a la 103, pero dependiendo de si la máquina es little o big endian, los bytes se almacenarían de diferente manera:

Little-endian (Como Intel)

| | | | | | |
|-----|-----|-----|-----|-----|-----|
| | 100 | 101 | 102 | 103 | |
| ... | DD | CC | BB | AA | ... |

Big-endian (Como Motorola)

| | | | | | |
|-----|-----|-----|-----|-----|-----|
| | 100 | 101 | 102 | 103 | |
| ... | AA | BB | CC | DD | ... |

En las imágenes de arriba, en donde se representan las posiciones de memoria 100, 101, 102 y 103 creciendo de izquierda a derecha, «parece» que la representación *big endian* es más natural, ya que el número AABBCDD lo podemos leer correctamente (ver figura), mientras que en la representación *little endian* parece que el número está al revés, o «patas arriba». Sin embargo, no hay nada que nos impida imaginarnos que las direcciones de memoria «crecen» de derecha a izquierda, y al observar la memoria de esta manera, la representación *little endian* «se ve natural» y es la *big endian* la que «parece» al revés, como se muestra en las figuras de abajo.

Little-endian

| | | | | | |
|-----|-----|-----|-----|-----|-----|
| | 103 | 102 | 101 | 100 | |
| ... | AA | BB | CC | DD | ... |

Big-endian

| | | | | | |
|-----|-----|-----|-----|-----|-----|
| | 103 | 102 | 101 | 100 | |
| ... | DD | CC | BB | AA | ... |

La información se representa por medio de caracteres, codificándose internamente en un alfabeto binario, es decir, en bits. Por tanto, a cada carácter le corresponde cierto número de bits. **Un BYTE es el número necesario de bits para almacenar un carácter.** Este número depende del código utilizado por el computador, siendo generalmente 8, por lo que habitualmente byte se utiliza como sinónimo de 8 bits u octeto. La capacidad de almacenamiento del computador o de un soporte de información (tal como un disco o una cinta magnética) se suele medir en bytes. Como el byte es una unidad relativamente pequeña, es usual utilizar múltiplos:

1 Kilobyte (o KB) = 2^{10} bytes = 1 024 bytes $\cong 10^3$ bytes
 1 Megabyte (o MB) = 2^{20} bytes = 1 048 576 bytes $\cong 10^6$ bytes
 1 Gigabyte (o GB) = 2^{30} bytes = 1 073 741 824 bytes $\cong 10^9$ bytes
 1 Terabyte (o TB) = 2^{40} bytes $\cong 10^{12}$ bytes
 1 Petabyte (o PB) = 2^{50} bytes $\cong 10^{15}$ bytes
 1 Exabyte (o EB) = 2^{60} bytes $\cong 10^{18}$ bytes

Los múltiplos anteriores (K, M, G, T, P y E) no sólo se utilizan con bytes, sino también con otras unidades internas de información. Así 1 Gb (o Gigabit) son 1 073 741 824 bits. En lo sucesivo utilizaremos una "b" para indicar bit, y una "B" para indicar byte.

Se entiende por **operaciones lógicas** funciones tales como comparar, seleccionar o copiar símbolos, ya sean numéricos o no numéricos.

REPRESENTACIÓN DE LA INFORMACIÓN EN EL ORDENADOR.

Un ordenador es una máquina que procesa información. La ejecución de un programa implica la realización de unos tratamientos, según especifica el programa (o conjunto ordenado de instrucciones), con los datos. Para que el ordenador ejecute un programa es necesario darle dos tipos de informaciones: **las instrucciones** que forman el programa y **los datos** con los que debe operar ese programa.

Dos de los aspectos más importantes que se presentan en Informática relacionados con la información son cómo "representarla" y cómo "materializarla" o "registrarla" físicamente. Normalmente la información se le suministra al ordenador en la forma usual que utilizamos los seres humanos; es decir, con ayuda de un alfabeto o conjunto de símbolos, que denominaremos "**caracteres**".

Los caracteres que constituyen el alfabeto suelen agruparse en cinco categorías:

1. **Caracteres alfabéticos:** son las letras mayúsculas y minúsculas del abecedario inglés: A, B, C, D, ..., Z, a, b, c, d, ..., z.
2. **Caracteres numéricos:** están constituidos por las diez cifras decimales: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.
3. **Caracteres especiales:** son los símbolos no incluidos en los grupos anteriores, siendo entre otros, los siguientes:)(*;/:;Ññ=!?.! "&>#<]Ç[SP. Con SP se representa el carácter o espacio en blanco, tal como el que separa dos palabras.
4. **Caracteres de control:** representan órdenes de control, como el carácter indicador de fin de línea o el carácter indicador de sincronización de una transmisión o de que se emita un pitido en un terminal, etc... Muchos de los caracteres de control son generados e insertados por el propio ordenador.
5. **Caracteres gráficos:** son símbolos o módulos con los que se pueden representar figuras (o iconos) elementales.

Al conjunto de los caracteres enumerados en los tres primeros puntos se les denomina caracteres alfanuméricos.

SISTEMAS DE NUMERACIÓN USUALES EN INFORMÁTICA.

Los ordenadores suelen efectuar las operaciones aritméticas utilizando una representación para los datos numéricos basada en el sistema de numeración base dos -sistema binario-.

También se utilizan los sistemas de numeración, preferentemente el octal y hexadecimal, para obtener códigos intermedios. Un número expresado en uno de estos dos códigos puede transformarse directa y fácilmente a binario y viceversa. Con ellos se simplifica la transcripción de números binarios y se está más próximo al sistema que utilizamos usualmente (el sistema decimal), por lo que a veces se utilizan como

paso intermedio en las transformaciones de decimal a binario y viceversa. Además, la realización electrónica de codificadores/decodificadores entre binario y un código intermedio es mucho menos compleja que entre binario y decimal.

El primer sistema de numeración del cual se tiene conocimiento fue el sistema egipcio. Posteriores a él son el romano, el maya, el chino, el indio, el árabe original hasta llegar al decimal actual.

EL SISTEMA DECIMAL

El sistema decimal es un sistema posicional, ya que el significado de un símbolo depende fundamentalmente de su posición relativa al símbolo coma (.), denominado *coma decimal*, que en caso de ausencia se supone colocada implícitamente a la derecha.

Utiliza como base el 10, que corresponde al número de símbolos que comprenden para la representación de cantidades; estos símbolos (también denominados dígitos) son:

0 1 2 3 4 5 6 7 8 9¹

Una determinada cifra, que se denominará *número decimal*, se puede expresar de la siguiente forma:

$$N^{\circ} = \sum_{i=-d}^n (\text{dígito})_i * (\text{base})^i$$

Donde:

- base = 10
- i = posición respecto a la coma
- d = n.º de dígitos a la derecha de la coma,
- n = n.º de dígitos a la derecha de la coma - 1,
- dígito = cada uno de los que componen el número

La fórmula responde al Teorema Fundamental de la Numeración.

El sistema decimal es un sistema posicional como ya hemos dicho, ya que el mismo dígito puede variar su valor de acuerdo a su posición.

Ej.:

| | |
|------|--------------|
| 1000 | mil |
| 100 | cien |
| 10 | diez |
| 1 | uno |
| 0,1 | un décimo |
| 0,01 | un centésimo |

¹ En todo sistema de numeración la base no aparece como dígito.

TEOREMA FUNDAMENTAL DE LA NUMERACIÓN

El teorema fundamental de la numeración dice:

"El valor en el sistema decimal de una cantidad expresada en otro sistema cualquiera de numeración, viene dado por la fórmula:

$$\dots + X_4 \cdot B^4 + X_3 \cdot B^3 + X_2 \cdot B^2 + X_1 \cdot B^1 + X_0 \cdot B^0 + X_{-1} \cdot B^{-1} + X_{-2} \cdot B^{-2} + X_{-3} \cdot B^{-3} + \dots"$$

donde **X** es el dígito y **B** la base.

Ejemplo:

Supongamos la cantidad $3221,03_4$ esta expresada en base 4 (ver subíndice al final de la cantidad), dicha base utiliza para representar cantidades los dígitos 0, 1, 2 y 3. ¿Cuál será el valor correspondiente en el sistema decimal?

$$3 \cdot 4^3 + 2 \cdot 4^2 + 2 \cdot 4^1 + 1 \cdot 4^0 + 0 \cdot 4^{-1} + 3 \cdot 4^{-2} =$$

$$3 \cdot 64 + 2 \cdot 16 + 2 \cdot 4 + 1 \cdot 1 + 0 \cdot 0,25 + 3 \cdot 0,0625 = 233,1875$$

El teorema aplicado a la inversa nos sirve para obtener el valor en una base cualquiera de un valor decimal, por medio de divisiones sucesivas por dicha base, como se verá más adelante.

EL SISTEMA BINARIO

Por razones técnicas, la mayoría de los circuitos electrónicos que conforman un ordenador solo puede detectar la presencia o ausencia de tensión en el circuito. Si a la presencia de tensión en un punto del circuito le asignamos el valor 1 y a la ausencia de la misma el valor 0 (a esta lógica se la denomina **lógica positiva**). Caso contrario la denominaremos **lógica negativa**.

Por las razones antes vistas, ya que el hardware por el momento solo reconoce estos dos estados fue necesario crear un sistema de numeración basado en estos dos valores (0, 1), al cual se lo denominó Binario, y cuya base por lo tanto es 2 (números de dígitos del sistema).

En computación cada dígito de un número representado en este sistema se denomina **bit** (contracción de **binary digit**).

Como múltiplos del bit hallamos:

- 8 bits \equiv **Byte** (palabra)² **B** (10110110)
- 1024 bytes \equiv 1 kilobyte **KB**
- 1024 KB \equiv 1 Megabyte **MB**
- 1024 MB \equiv 1 Gigabyte **GB**
- 1024 GB \equiv 1 Terabyte **TB**
- 1024 TB \equiv 1 Exabyte **EB**

² La idea de palabra queda de las antiguas computadoras con palabras de 8 bits, hoy existen máquinas cuya palabra es de 16, 32, 64 bits.

Dos cosas a tener en cuenta:

- La B de byte es siempre mayúscula, ya que Kb significa Kbit unidad utilizada en las memorias.
- En el sistema de numeración decimal los múltiplos son potencias 10 ($1K \equiv 1000$ unidades y $1M \equiv 1000 K$), en el binario es $2^{10} = 1024$.

OPERACIONES CON BINARIOS

Tanto la suma como la multiplicación son semejantes a la decimal con la diferencia que se maneja solo dos dígitos, sus tablas de operación se pueden observar en los siguientes esquemas

Suma

| | | |
|---|---|----|
| + | 0 | 1 |
| 0 | 0 | 1 |
| 1 | 1 | 10 |

Multiplicación

| | | |
|---|---|---|
| * | 0 | 1 |
| 0 | 0 | 0 |
| 1 | 0 | 1 |

Ejemplos

| | | |
|-------|---------------|---------|
| 1 1 1 | 1 1 | Acarreo |
| | 1 1 0 0 1 | 25 |
| + | 1 0 1 0 1 1 | + 43 |
| | 1 0 0 0 1 0 0 | 68 |

| | |
|------------|----------------|
| 1 1 | Acarreo |
| 1 1 0' 1 0 | 6,50 |
| + | 1 1 0 1' 0 1 |
| | 1 0 0 1 1' 1 1 |
| | 19,75 |

| | |
|-----------|-------------------|
| 1 1 0 0 1 | 25 |
| * | 1 0 0 1 1 |
| | 1 1 0 0 1 |
| | 1 1 0 0 1 |
| | 1 1 0 0 1 |
| | 1 1 0 0 1 0 0 |
| | 1 1 1 0 1 1 0 1 1 |
| | 475 |

La resta como la división son procesos que la unidad de cálculo del ordenador no realiza por lo tanto no lo vamos a ver en forma directa.

EL SISTEMA OCTAL

Es un sistema cuya base es el número 8, es decir, utiliza 8 símbolos para la representación de un valor cualquiera. Estos símbolos son:

0 1 2 3 4 5 6 7

Este es un sistema también posicional, de aritmética muy similar al decimal. Su utilización comenzó como sistema de salida de las computadoras ya que para representar un valor la cantidad de símbolos que necesita es menor que el binario y la conversión entre ambos sistemas es muy sencilla de implementar.

EL SISTEMA HEXADECIMAL

Es un sistema cuya base es el número 16, es decir, utiliza 16 símbolos para la representación de un valor cualquiera. Estos símbolos son:

0 1 2 3 4 5 6 7 8 9 A B C D E F

Este es otro sistema posicional, de característica similar al octal. Su uso fue adoptado por idénticas razones que el octal.

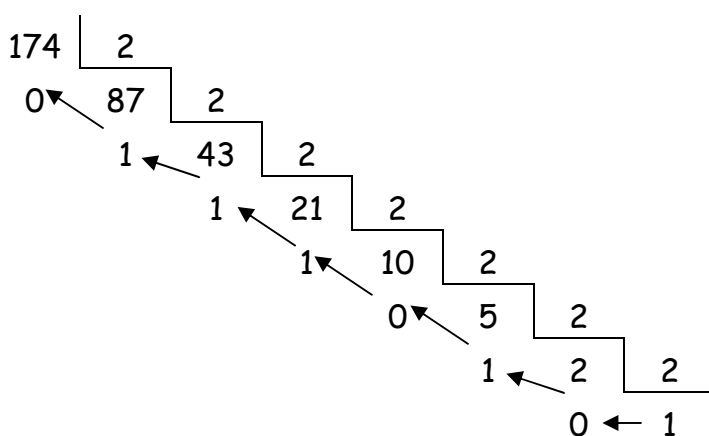
CONVERSIÓN ENTRE LOS DISTINTOS SISTEMAS

Se denomina así la transformación de un valor en un sistema al equivalente en otro sistema.

Conversión decimal a binario

Para convertir un número decimal entero a binario, este debe ser dividido por dos y repetir el proceso con sus cocientes hasta que el cociente tome el valor 1. La unión de todos restos escritos en orden inverso encabezados por el último cociente, nos dará el valor expresado en binario.

Ej. : Convertir el número 174 a binario



$$174_{10} = 10101110_2$$

Para convertir una fracción decimal a binario, esta fracción debe ser multiplicada por dos y tomamos la parte entera del resultado, repetimos el proceso con la parte fraccionaria del resultado anterior, dándonos una nueva parte entera, y así sucesivamente hasta que la parte fraccionaria se haga 0 (cero) o que tengamos suficientes decimales que nos permita estar debajo de un determinado error.

Ej. : Convertir el número 0,90625 a fracción binaria

$$0,90625 * 2 = 1,8125$$

$$0,8125 * 2 = 1,625$$

$$0,625 * 2 = 1,25$$

$$0,25 * 2 = 0,5$$

$$0,5 * 2 = 1,$$

$$0,90625_{10} = 0,11101_2$$

Ej. : Convertir el número 0,64037 a fracción binaria

$$0,64037 * 2 = 1,28074$$

$$0,28074 * 2 = 0,56148$$

$$0,56148 * 2 = 1,12296$$

$$0,12296 * 2 = 0,24592$$

$$0,24592 * 2 = 0,49184$$

$$0,49184 * 2 = 0,98368$$

$$0,98368 * 2 = 1,96736$$

$$0,96736 * 2 = 1,93472$$

$$0,93472 * 2 = 1,86944$$

$$0,86944 * 2 = 1,73888$$

$$0,64037_{10} = 0,1010001111_2$$

El error en el valor es $\varepsilon \leq 2^{-10} \Rightarrow \varepsilon \leq 0,001$. Esto es así porque hemos obtenido 10 unidades binarias, de querer mejorar la precisión deberemos obtener un mayor número de fracciones binarias.

Pase a binario las siguientes fracciones decimales con $\varepsilon \leq 2^{-10}$: 0,63965 y 0,064062.

Si se desea convertir un número que tiene parte entera y decimal a binario, se deberá operar cada parte por separado como ya se ha visto, y luego obtener la suma de los resultados.

Por ejemplo:

$$174,90625_{10} = 10101110,11101_2$$

Conversión binario a decimal

Para realizar esta conversión se utiliza como base el teorema fundamental de la numeración.

El método práctico consiste en multiplicar cada uno de los términos por potencias crecientes de 2 a partir de la coma decimal y hacia la izquierda, y realizar la suma de las operaciones.

Por ejemplo:

Pasar a decimal el binario 10101110_2

| | | | | | | | |
|---|---------------|----------------|-----------------|---------------|---------------|---------------|---------------|
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| | | | | | | | |
| | | | | | | → | $0 * 2^0 = 0$ |
| | | | | | → | $1 * 2^1 = 2$ | |
| | | | | → | $1 * 2^2 = 4$ | | |
| | | | → | $1 * 2^3 = 8$ | | | |
| | | → | $0 * 2^4 = 0$ | | | | |
| | → | $1 * 2^5 = 32$ | | | | | |
| → | $0 * 2^6 = 0$ | | | | | | |
| | | → | $1 * 2^7 = 128$ | | | | |
| | | | | | | | 174 |

$10101110_2 = 174_{10}$

En los casos de números que posean parte entera y decimal se recomienda el uso del teorema fundamental de la numeración.

Ej.: Convertir $1101,011_2$ a base 10

Para pasar a base 10 deberemos hacer:

$$1 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0 + 0 * 2^{-1} + 1 * 2^{-2} + 1 * 2^{-3} =$$

$$1 * 8 + 1 * 4 + 0 + 1 * 1 + 0 + 1 * 0,25 + 1 * 0,125 =$$

$$8 + 4 + 0 + 1 + 0 + 0,25 + 0,125 = 13,375$$

$$1101,011_2 = 13,375_{10}$$

Conversión octal a binario

Al ser la base del octal (8) potencia de la base binaria (2^3), la transformación de una base a la otra se hace en forma directa dígito a dígito. Cada dígito octal será reemplazado por 3 dígitos binarios (3 por ser la potencia que relaciona ambas bases), según la tabla que tenemos a continuación.

| Octal | Binario |
|-------|---------|
| 0 | 000 |
| 1 | 001 |
| 2 | 010 |
| 3 | 011 |
| 4 | 100 |
| 5 | 101 |
| 6 | 110 |
| 7 | 111 |

Ej.:

Convertir a binario el número $276,534_8$

| | | | | | |
|-----|-----|------|-----|-----|-----|
| 2 | 7 | 6, | 5 | 3 | 4 |
| 010 | 111 | 110, | 101 | 011 | 100 |

$$276,534_8 = 10111110,1010111_2$$

Como se puede ver los ceros al comienzo se han quitado, igual que los ceros que se hallan a la derecha de la coma (ya que no tienen ningún sentido).

Conversión binario a octal

Esta conversión es similar a la anterior, pero cada tres símbolos binarios corresponde uno octal. Para realizar correctamente esta conversión el **número de dígitos** a la derecha de la coma decimal debe ser múltiplo de 3 si no lo fuera deberá **agregarse al final** del número tantos ceros como sea necesario. Idéntico caso será a la izquierda de la coma, en dicho caso los ceros se agregan al principio del número.

Ej.

Convertir el binario $10101011,0011$ a octal.

| | | | | |
|-----|-----|------|-----|-----|
| 010 | 101 | 011, | 001 | 100 |
| 2 | 5 | 3, | 1 | 4 |

0 cero agregado al número para permitir la correcta conversión.

$$10101011,0011_2 = 253,14_8$$

Conversión hexadecimal a binario

Por idénticas razones que el caso anterior ($16 = 2^4$), la transformación de una base a la otra se hace en forma directa dígito a dígito. Cada dígito hexadecimal será reemplazado por 4 dígitos binarios (4 por ser la potencia que relaciona ambas bases), según la tabla que tenemos a continuación.

| Hexadecimal | Binario |
|-------------|---------|
| 0 | 0000 |
| 1 | 0001 |
| 2 | 0010 |
| 3 | 0011 |
| 4 | 0100 |
| 5 | 0101 |
| 6 | 0110 |
| 7 | 0111 |

| Hexadecimal | Binario |
|-------------|---------|
| 8 | 1000 |
| 9 | 1001 |
| A | 1010 |
| B | 1011 |
| C | 1100 |
| D | 1101 |
| E | 1110 |
| F | 1111 |

Ej.:

Convertir a binario el número $5A8,39C_{16}$

| | | | | | |
|------|------|-------|------|------|------|
| 5 | A | 8, | 3 | 9 | C |
| 0101 | 1010 | 1000, | 0011 | 1001 | 1100 |

$$5A8,39C_{16} = 10110101000,0011100111_2$$

Como se puede ver otra vez los ceros al comienzo se han quitado, igual que los ceros que se hallan a la derecha de la coma (ya que no tienen ningún sentido).

Conversión binario a hexadecimal

Esta conversión es similar a la conversión a octal, pero en lugar de tres, serán cuatro símbolos binarios los que corresponde a un hexadecimal. Para realizar correctamente esta conversión el **número de dígitos** a la derecha de la coma decimal debe ser múltiplo de 4 si no lo fuera deberá **agregarse al final** del número tantos ceros como sea necesario. Idéntico caso será a la izquierda de la coma, en dicho caso los ceros se agregan al principio del número.

Ej.

Convertir el binario $1010101011,0011$ a hexadecimal.

| | | | | |
|------|------|-------|------|------|
| 0010 | 1010 | 1011, | 0011 | 1000 |
| 2 | A | B, | 3 | 8 |

0 cero agregado al número para permitir la correcta conversión.

$$1010101011,0011_2 = 2AB,38_{16}$$

Conversión decimal a octal o hexadecimal

Para cualquiera de estos dos casos se hará en forma similar a la explicada para convertir de decimal a binario 1.7.4. Pero se deberá tener en cuenta que la base ya no es 2, sino 8 o 16 según corresponda. (Dividir por 8 o 16)

Conversión octal o hexadecimal a decimal

Para cualquiera de estos dos casos se deberá usar el teorema fundamental de la numeración, teniendo en cuenta base que corresponda (8 o 16 según el caso).

Conversión octal a hexadecimal o hexadecimal a octal.

Estas conversiones no son posibles en una forma directa. Para realizar cualquiera de ellas se deberá usar el pasaje a otra base como paso intermedio.

Por ejemplo $\text{octal} \Leftrightarrow \text{decimal} \Leftrightarrow \text{hexadecimal}$

$\text{octal} \Leftrightarrow \text{binario} \Leftrightarrow \text{hexadecimal}$

Se recomienda como metodología de trabajo esta última, porque al ser las operaciones de conversión más sencillas disminuye la probabilidad de error. Además no existe la posibilidad de errores de redondeo.

REPRESENTACIÓN DE NÚMEROS ENTEROS

Existen 4 formas de representar un número entero en un ordenador (todos en sistema binario), ellas son

- Módulo y signo
- Complemento a 1 (C-1)
- Complemento a 2 (C-2)
- Exceso a 2 elevado a la N -1

En todos los casos se considera que tenemos un número limitado de dígitos para cada elemento numérico. El número de dígitos disponibles lo representa N (8, 16, 32, 64 o sea 1, 2, 3, 4... Bytes).

Módulo y signo.

En este método se utiliza el primer bit a la izquierda como signo, 0 si es positivo y uno si es negativo. Los restantes (7, 15, etc.), representan el módulo

Por ejemplo

| | Signo | Mantisa |
|----------------------------------|-------|-----------------|
| 19 se representa en 8 bits como | 0 | 0010011 |
| -19 | 1 | 0010011 |
| 19 se representa en 16 bits como | 0 | 000000000010011 |
| -19 | 1 | 000000000010011 |

El conjunto de valores que se puede representar en un método determinado se conoce como **rango** de la representación. Para módulo y signo el rango de representación para N dígitos es:

$$- 2^{N-1} + 1 \leq x \leq 2^{N-1} - 1$$

Para 1 Byte (8 bits) es

$$-127 \leq x \leq 127$$

Para 2 Byte (16 bits) es

$$-32767 \leq x \leq 32767$$

Para 4 Byte (32 bits) es

$$-2147483647 \leq x \leq 2147483647$$

Este método tiene la ventaja de poseer un rango simétrico, pero la desventaja de poseer dos representaciones para el número 0

Complemento a 1 (C-1).

Para representar un número positivo es igual al método de MS. Pero en el caso de los negativos, se obtiene complementando al positivo (cambiando 1 por 0 y viceversa)

Por ejemplo

| | Signo | Mantisa |
|----------------------------------|-------|-----------------|
| 19 se representa en 8 bits como | 0 | 0010011 |
| -19 | 1 | 1101100 |
| 19 se representa en 16 bits como | 0 | 000000000010011 |
| -19 | 1 | 11111111101100 |

Para complemento a 1 el rango de representación para N dígitos es:

$$- 2^{N-1} + 1 \leq x \leq 2^{N-1} - 1$$

Para 1 Byte (8 bits) es

$$-127 \leq x \leq 127$$

Para 2 Byte (16 bits) es

$$-32767 \leq x \leq 32767$$

Para 4 Byte (32 bits) es

$$-2147483647 \leq x \leq 2147483647$$

Este método presenta iguales ventajas y desventajas que el anterior.

Complemento a 2 (C-2)

Este método es similar al anterior, la representación de los números positivos es igual a la anterior, pero los negativos se obtienen en dos pasos:

- Se complementa a 1
- Al resultado se le suma 1

Por ejemplo

| | | | |
|---------------------------------|---|---------|-----|
| 19 se representa en 8 bits como | 0 | 0010011 | |
| -19 | 1 | 1101100 | C-1 |
| | | +_____1 | |
| -19 | 1 | 1101101 | C-2 |

Para complemento a 2 el rango de representación para N dígitos es:

$$- 2^{N-1} \leq x \leq 2^{N-1} - 1$$

Para 1 Byte (8 bits) es

$$-128 \leq x \leq 127$$

Para 2 Byte (16 bits) es

$$-32768 \leq x \leq 32767$$

Para 4 Byte (32 bits) es

$$-2147483648 \leq x \leq 2147483647$$

Presenta las siguientes ventajas. Tiene una única representación para 0, la segunda es que en lugar de hacer $A - B$, puedo hacer $A + B_{C-2}$. La unidad aritmético lógica del microprocesador solo suma, **no resta**.

Exceso a 2 elevado a la N -1

En este método no hay bit de signo, todos los bits se utilizan para representar el valor del número más el exceso, que para N bits viene dado por 2^{N-1} , que para una representación de 8 bits es 128.

Para obtener un número en un exceso dado, se realiza la suma algebraica del exceso más el número. Solo se pueden representar valores en módulo menores o iguales al exceso.

Ej.

| | |
|------------------|------------|
| Exceso 128 | 10000000 |
| 19 | + 00010011 |
| 19 en exceso 128 | 10010011 |

Por ejemplo

| | | |
|---------------------------------|---|---------|
| 19 se representa en 8 bits como | 1 | 0010011 |
| -19 | 0 | 1101101 |

En este método el 0 tiene única representación, el rango de representación es asimétrico.

Para complemento a 2 el rango de representación para N dígitos es:

$$- 2^{N-1} \leq x \leq 2^{N-1} - 1$$

Para 1 Byte (8 bits) es

$$-128 \leq x \leq 127$$

Para 2 Byte (16 bits) es

$$-32768 \leq x \leq 32767$$

Para 4 Byte (32 bits) es

$$-2147483648 \leq x \leq 2147483647$$

La representación en exceso para un número cualquiera es igual a la representación en complemento a dos pero el valor del primer bit de la izquierda esta invertido.

DESBORDAMIENTO (OVERFLOW)

Este hecho se puede producir cuando se suman dos números en un método de representación y el resultado no puede ser representado por el método, dándonos un resultado erróneo. Para el ejemplo usaremos la notación de MS

Ej.

| | | |
|------|-------------------|-----|
| 52 | 0 0 1 1 0 1 0 0 | 52 |
| + 97 | + 0 1 1 0 0 0 0 1 | 97 |
| 149 | 1 0 0 1 0 1 0 1 | -21 |

PRECISION FINITA DE LA INFORMACIÓN

Muchos estudiantes consideran que el ordenador puede trabajar con números con cantidades de cifras infinitamente grandes. Este preconcepto es uno de los más erróneos que se puede detectar en el alumno.

Todo ordenador cuenta con un número finito de Bytes para poder almacenar una cifra. Este número puede ser de 1, 2, 4, 6, 8, 10 Bytes, pero nunca infinito. Por lo tanto solo se podrá ingresar, procesar, almacenar y extraer un rango de valores. Por ejemplo para números enteros se utiliza como máximo 4 Bytes (32 bits), siendo el rango de representación entre -2147483648... 2147483647.

Coma Flotante

Este método nace de la necesidad de representar números reales o enteros con un rango mayor que el dado por los otros métodos.

En su representación se utiliza la representación matemática

$$N^O = \text{mantisa} * \text{base}^{\text{exponente}}$$

Por ejemplo

$$79436.54 = 0,7943654 * 10^5$$

A este proceso se lo denomina normalización.

Para estos números se utilizan explicaremos dos formas de representación simple y doble precisión, pero existen otros formatos como real, extended, o comp.

Para simple precisión se utiliza 32 bits (4 Bytes), en el segundo caso 64 bits (8 Bytes). (Todos los elementos en computación se comienzan a numerar por 0)

El esquema en ambos casos es como se ve abajo.

| | Simple Precisión | | | | Doble Precisión | | |
|-----------|------------------|------------|----------|--|-----------------|------------|----------|
| | C. de bits | B. Inicial | B. Final | | C. de bits | B. Inicial | B. Final |
| Signo | 1 | 31 | | | 1 | 63 | |
| Exponente | 8 | 23 | 30 | | 11 | 52 | 62 |
| Mantisa | 23 | 0 | 22 | | 52 | 0 | 51 |

Ejemplos de Conversión de Decimal a Flotante

57 a Flotante

1) Paso 57 a Binario

$$57 \Rightarrow 111001$$

2) Normalizo el binario

$$111001 \Rightarrow 0,111001 * 2^6$$

3) Paso el exponente a binario

$$6 \Rightarrow 110$$

4) Si trabajo en Simple Precisión (SP) lo expreso como excedente a 10000000 (por los 8 bits), si es en Doble Precisión como excedente a 10000000000 (por los 11 bits).

EL **exponente** nos queda así.

SP 10000110

DP 10000000110

5) Como el número es positivo el bit de signo es 0

El número queda estructurado de la siguiente manera

| | | | |
|----|-------|-----------|---------|
| | Signo | Exponente | Mantisa |
| SP | 0 | 10000110 | 111001 |

Debería agregar 0 hasta completar los 24 bits

El número en cuestión nos queda

0100 0011 0111 0010 0000 0000

7) Lo paso a HEXADECIMAL y nos queda

$$4372_{16}$$

En el caso de - 56

8) Como el número es negativo el bit de signo es 1

El número queda estructurado de la siguiente manera

| | | | |
|----|-------|-----------|---------|
| | Signo | Exponente | Mantisa |
| SP | 1 | 10000110 | 111001 |

Debería agregar 0 hasta completar los 24 bits

El número en cuestión nos queda

1100 0011 0111 0010 0000 0000

9) Lo paso a HEXADECIMAL y nos queda

$$C372_{16}$$

Ejemplo de exponente negativo

El número 0,13671875 repito los pasos anteriores.

Paso a binario

$$0,13671875 \Rightarrow 0,00100011$$

Normalizo

$$0,00100011_2 \Rightarrow 0,100011_2 * 2^{-2}$$

Paso el modulo de la potencia a Binario

$$2 \Rightarrow 10_2$$

Si trabajo en Simple Precisión (SP) lo expreso como excedente a 10000000 EL **exponente** nos queda así.

$$SP \quad 01111110$$

Como el número es positivo el bit de signo es 0

El número queda estructurado de la siguiente manera

| | Signo | Exponente | Mantisa |
|----|-------|-----------|---------|
| SP | 0 | 01111110 | 100011 |

Debería agregar 0 hasta completar los 24 bits

El número en cuestión nos queda

$$0011 \ 1111 \ 0100 \ 0110$$

(no se completó con ceros porque su representación en Hexadecimal son 0 que no afectan el número final)

Lo paso a HEXADECIMAL y nos queda

$$3F46_{16}$$

Si el número fuera negativo el bit de signo es 1

El número queda estructurado de la siguiente manera

| | Signo | Exponente | Mantisa |
|----|-------|-----------|---------|
| SP | 1 | 01111110 | 100011 |

Debería agrega 0 hasta completar los 24 bits

El número en cuestión nos queda

$$1011 \ 1111 \ 0100 \ 0110$$

Lo paso a HEXADECIMAL y nos queda

$$BF46_{16}$$

Si el número (-0,13671875) quisiéramos expresarlo en flotante de 64 bits, el único cambio que tendríamos sería el exponente que ya no tiene 8 bits sino 11 bits quedándonos.

El número queda estructurado de la siguiente manera

| | Signo | Exponente | Mantisa |
|----|-------|------------|---------|
| SP | 1 | 0111111110 | 100011 |

El número en cuestión nos queda

$$1011 \ 1111 \ 1110 \ 1000 \ 1100$$

Lo paso a HEXADECIMAL y nos queda

$$BFE8C_{16}$$

Como se puede ver el mismo número según se represente en 32 o en 64 bits

| | 32 bits | 64 bits |
|-------------|----------|------------------|
| -0,13671875 | BF460000 | BFE8C00000000000 |

Los ceros a la izquierda no son representativos, pueden o no escribirse.

Este método de representación tiene sus rangos de representación los cuales **no** incluyen el número 0 (cero). Se puede representar números muy próximos a 0 pero no incluye este número.

El módulo mayor que se puede expresar en doble precisión es $1,710 * 10^{308}$, con una precisión de 15 a 16 cifras (ver transformación de fracciones decimales a binarios). El número más próximo a cero será $1 * 10^{-309}$. El módulo mayor que se puede expresar en punto flotante (extended) es $1,10 * 10^{4932}$.

REPRESENTACIÓN INTERNA DE LA INFORMACIÓN: Codificación alfanumérica

Cada vez que presionamos una tecla cualquiera en nuestra computadora, esta convierte el carácter presionado en un conjunto bits. Para esta transformación se utilizaron y se utilizan distintos códigos.

El primero fue un código de 6 bits denominado FIELDATA. Es código fue reemplazado por el ASCII (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange) que era un código de 7 bits (tenía 128 caracteres posibles), luego aparece el EBCDIC que fue el primer código de 8 bits por último aparece para el ambiente de PC el ASCII extendido que también es de 8 bits (256 caracteres).

Tabla de conversión

| Decimal | Binario | Octal | Hexadecimal |
|---------|---------|-------|-------------|
| 0 | 0000 | 00 | 0 |
| 1 | 0001 | 01 | 1 |
| 2 | 0010 | 02 | 2 |
| 3 | 0011 | 03 | 3 |
| 4 | 0100 | 04 | 4 |
| 5 | 0101 | 05 | 5 |
| 6 | 0110 | 06 | 6 |
| 7 | 0111 | 07 | 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 |
| 31 | 11111 | 37 | 1F |
| 32 | 100000 | 40 | 20 |

SISTEMAS DE NUMERACIÓN

Práctica

1 - Pasar a base 10 los siguientes números, de las bases indicadas:

1101_2 $0,101_2$ $101,11_2$ $1,0111_2$ 753_8

$0,63_8$ $17,134_8$ $3A_{16}$ $0,FF_{16}$ $A5,3B_{16}$

2 - Pasar los siguientes números de base 10 a la base indicada:

$39 \Rightarrow_2$ $0,525 \Rightarrow_2$ $23,945 \Rightarrow_2$ $123 \Rightarrow_8$

$3,1 \Rightarrow_8$ $0,14 \Rightarrow_8$ $1068 \Rightarrow_{16}$ $61,6 \Rightarrow_{16}$

3 - Pasar el siguiente decimal a la base indicada con un error menor o igual al indicado

| Número | Base | Error |
|--------|------|--------|
| 0,267 | 2 | 0,001 |
| 52,38 | 2 | 0,0001 |
| 129,64 | 2 | 0,1 |
| 163,97 | 8 | 0,0001 |
| 954,62 | 16 | 0,0001 |

4 - Pasar a las bases indicadas usando propiedad de base de potencia de otra base:

$32_8 \Rightarrow_2$ $F1_{16} \Rightarrow_8$ $F1_{16} \Rightarrow_2$

$73_8 \Rightarrow_{16}$ $1010_2 \Rightarrow_{16}$ $10,10_2 \Rightarrow_8$

5 - Realizar las siguientes sumas:

1010_2 1001_2 1110_2
 $+ 0101_2$ $+ 0110_2$ $+ 1010_2$

7354_8 $F1E5_{16}$ 3231_4
 $+ 1123_8$ $+ ABC1_{16}$ $+ 2123_4$

6 - Realizar las siguientes restas:

$$\begin{array}{r} \text{F91F}_{16} \\ -0101_{16} \end{array}$$

$$\begin{array}{r} 0334_8 \\ -0137_8 \end{array}$$

$$\begin{array}{r} 1060_8 \\ -1776_8 \end{array}$$

7 - Realizar las siguientes operaciones por Complemento a la Base

$$\begin{array}{r} 10011101_2 \\ -00110011_2 \end{array}$$

$$\begin{array}{r} 01110101_2 \\ -00011111_2 \end{array}$$

$$\begin{array}{r} 00100011_2 \\ -00011001_2 \end{array}$$

8 - Realizar las siguientes restas en base 2. Los números tienen signo.

$$\begin{array}{r} 01000 \\ -00101 \\ \hline \end{array}$$

$$\begin{array}{r} 11001 \\ -00111 \\ \hline \end{array}$$

$$\begin{array}{r} 00110 \\ -11000 \\ \hline \end{array}$$

9 - Realizar los siguientes productos.

$$\begin{array}{r} 0018_{16} \\ \times 100_{16} \\ \hline \end{array}$$

$$\begin{array}{r} 047_8 \\ \times 010_8 \\ \hline \end{array}$$

$$\begin{array}{r} 0018_{18} \\ \times 010_{18} \\ \hline \end{array}$$

10 - Escribir con notación exceso 10000000₂

$$1010_2$$

$$-F1_{16}$$

$$3014_8$$

$$-1100_2$$

$$-513_8$$

$$-37_{16}$$

11 - Escribir como complemento a Dos (en 16 bits):

$$35_{10}$$

$$-47_{10}$$

$$F1_{16}$$

$$-16_{16}$$

12 - Escribir como complemento a Dos (en 32 bits):

$$-93_{10}$$

$$-FF_{16}$$

$$-10_{10}$$

$$-31_{10}$$

- $F3_{16}$ - 16_{16}

13 - Pasar a base 10 los números (16 bits complemento a dos):

1) 1000000000101000 2) 1110100000010101

3) 1001111011010111 4) 1000000000010101

14 - Pasar a base 10 los siguientes números expresados como punto fijo sin signo (16 bits)

1000000000101000 0110100000010101

1001111011010111 0000000000010101

15 - Escribir con notación exceso 10000000 2

1010_2 - $F1_{16}$ 3014_5

33_4 - 513_6 - 37_{16}

16 - Escribir en base 2 y operar por complemento a la base

| | | |
|-------------|-------------|--------------|
| 5349_{10} | $F1F0_{16}$ | -3511_{10} |
| - | + | - |
| $317F_{16}$ | -34312 | $39F1_{16}$ |
| <hr/> | <hr/> | <hr/> |
| | 10 | |

17 - Escribir como complemento a Dos (en 16 bits):

35_{10} - 47_{10} $F1_{16}$ - 16_{16}

18 - Escribir como complemento a Dos (en 32 bits):

- 93_{10} - FF_{16} - 10_{10} - 31_{10}

- $F3_{16}$ - 16_{16}

19 - Expresar en base 10 los siguientes números dados en formato de Punto Flotante

$35A1F$ $93900D$ ECF $3ED$

20 - Pasar a base 10 los números (16 bits complemento a dos):

1) 1000000000101000 2) 0110100000010101

3) 1001111011010111 4) 0000000000010101

Realizar 1) + 2) y 1) - 4)

21 - Pasar a base 10 los siguientes números expresados como punto fijo sin signo (16 bits)

1000000000101000 0110100000010101

1001111011010111 0000000000010101

22 - Pasar a Punto Flotante:

| | | | |
|---------|---------------|-----------|--------------|
| 39 | $0,0103$ | $9F1$ | $9F3,61$ |
| -5826 | $-0,00002103$ | $-74F28B$ | $-0,002A359$ |

23 - Decir que número decimal, representa el siguiente número expresado como Punto Flotante

9 E C 1 9 3 5 F ₁₆

C D 9 4 0 1 0 3 ₁₆

3 E A C 1 0 0 0 ₁₆

A E 8 F 5 0 0 0 ₁₆

