

**INSTITUTO POLITÉCNICO NACIONAL**

**Unidad Profesional Interdisciplinaria de Ingeniería y Ciencias Sociales y Administrativas**

**Ingeniería en Informática**

**Arquitectura y Organización de Computadoras**

**Unidad Aritmética Lógica**

**2NM31**

**Díaz Álvarez Eduardo**

**Ciudad de México. 29 / 04 / 2020.  
Profesor Velasco Contreras Jose Antonio**

**Unidad Aritmética Lógica**

Una expresión binaria como 01111000 puede tener en el computador distintos significados: Un número binario sin signo Un número binario con signo Un número BCD Un carácter ASCII Una instrucción Una dirección El significado que se le dé dependerá del tipo de operación que se le imponga en el desarrollo de una tarea.

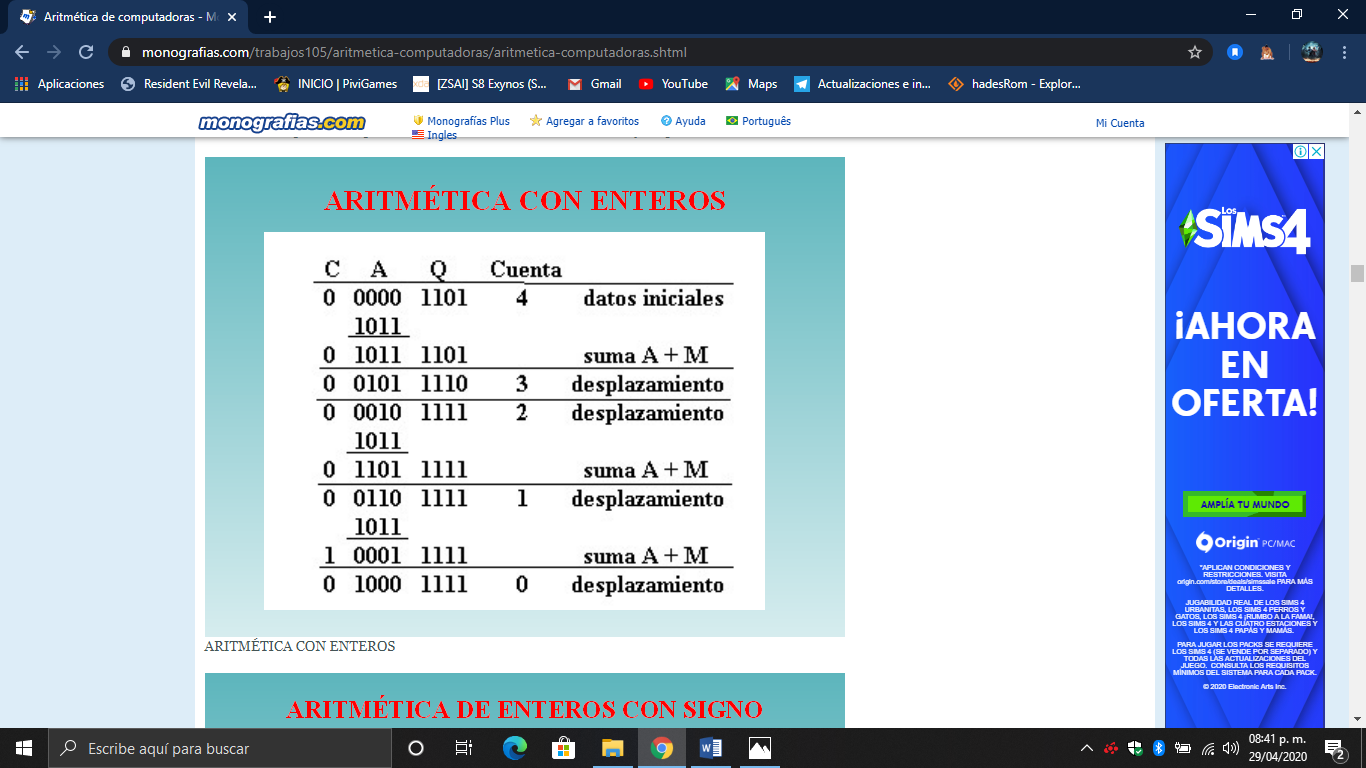
Si 01111000 es interpretado como un número binario sin signo su valor en base decimal sería 120. El algoritmo de conversión es como sigue: Para el número binario su valor es: es decir:

En la representación del número binario con signo, el bit más significativo del número binario es un bit de signo: si dicho bit es 0 el número es positivo y si es 1, el número es negativo. La magnitud de un número binario con signo positivo se determina utilizando el mismo algoritmo empleado anteriormente para números binarios sin signo. Así por ejemplo el número 00010010 corresponde en base decimal a +18.

Para representar un número binario negativo, se cuentan con dos notaciones: la notación del complemento a dos y la notación signo magnitud. Para obtener el negativo de un número positivo en notación complemento a dos, se complementan todos los bits y se suma un 1 al bit menos significativo.

Por ejemplo, el número positivo 01110110 tiene por complemento a dos el número 10001010 que es negativo porque su bit más significativo es 1.

El complemento 2 de este número es 01110110. Al aplicar el algoritmo anterior se obtiene la magnitud decimal 118. El número negativo es por tanto 118. De hecho, el negativo de un número binario con signo es su complemento a dos. Sin embargo, hay que señalar que la notación Complemento a Dos no es la única representación para los números binarios con signo. Otra notación muy conocida es la de signo magnitud.

En la notación signo magnitud, los números positivos se escriben de la misma manera que los números en notación complemento a dos. Por ejemplo, el número +118 se escribe como 01110110, donde el bit más significativo es el signo y los bits restantes representan la magnitud. Para determinar el correspondiente número negativo en esta notación, basta con cambiar el bit de signo, así el número 118 es 11110110, la magnitud permanece igual. Compárese con la representación del número 118 en complemento a dos.

Esta diferencia de notación en los números binarios con signo también se observa en los algoritmos utilizados para realizar distintas operaciones aritméticas. Por ejemplo, para sumar 7 = 1111 más +5 = 0101 en la notación signo magnitud hay que separar los signos y las magnitudes. Si los signos son iguales, las magnitudes se suman; si los signos son diferentes, la magnitud más pequeña debe restarse de la magnitud más grande y el resultado se acompaña del signo de la mayor magnitud: 1010.

ARITMÉTICA CON ENTEROS En cambio para la notación complemento a dos, la realización de la suma aritmética emplea el mismo algoritmo utilizados en números decimales. Ejemplos: (-7) + (+5) 1001 0101 1110 = -2 Acarreo = 0

(-7) + (-6) 1001 1010 0011 = 3 Acarreo = 1 En la última suma, el acarreo no está junto con el resultado, normalmente se ignora.

Entonces ocurre un desborde en el resultado de la operación y la computadora lo consigna como acarreo en alguno de sus registros internos. Es usual reconocer un desborde en el resultado si el signo de este es contrario a lo esperado.

representación de Números Racionales

Para la representación de los números Racionales existen dos métodos muy conocidos como el del punto fijo, y la representación en punto flotante.

1) Punto Fijo

El sistema usa palabras divididas en 3 campos:

Signo Parte del número precedente al punto binario Parte posterior al pto. Binario Desventajas: sólo se puede representar una pequeña cantidad de números. En nuestro caso, la palabra de 32 se divide en campos de 1, 15 y 16 bits, respectivamente, y los números están en el rango: 2-16 ≤ x < 215

Nota: raramente usada hoy en aplicaciones científicas.

2) Punto Flotante

Con el sistema de Punto Flotante, se resuelve el problema del punto fijo, ya que amplía el rango. Los números que conocemos como escritos en notación científica o notación exponencial se asemejan a la notación de punto flotante.

Un sistema de números en punto flotante es un subconjunto F⊂ ℜ de números reales, cuyos elementos x, x ∈ R-{0} se pueden escribir como:

x = ± m ×10E , con 1≤ m <10

Otra forma equivalente:

x = ± 0. d1d2. . . d t ×10E

Los parámetros que caracterizan el sistema de números flotantes de base diez, son:

• La base B

• La mantisa m, que representa a la parte fraccionaria del número

• El exponente E, que varía entre dos cotas: Emin ≤ E ≤ Emax

• La precisión t referida a la cantidad de dígitos d i donde: 0 ≤ di ≤ B-1

En los computadores se prefiere la notación de base 2:

x = ± m × 2E , con 1≤ m < 2

Para una palabra de memoria de 32 bits la representación se hace en 3 campos: 1 bit para el signo; 8 bits para el exponente E; y 23 bits para la mantisa.

El campo de la mantisa m puede representar los 23 bits que denominaremos:

b0 b1 b2 ...b22.

Si las cifras que siguen a b22 , o sea, b23, b24, ... no son todas 0, esta representación del número flotante x , designado fl (x), no es exacta, sino aproximada.

Sin embargo fl (x) puede ser expresado en forma aproximada mediante dos técnicas: truncamiento y redondeo (cfr.7 )

2.3.3 Definición de Número en Punto Flotante

Si un número x puede ser almacenado exactamente en el computador usando la representación de

punto flotante con b23 = b24 = .... = 0 , se llama número en punto flotante, y se lo designa fl (x)

Ejemplo 1:

x = =

(10

(10

2

11

101.1 (2 Este cociente puede multiplicarse y dividirse por 22, de modo que aparezca el primer dígito distinto de cero a la izquierda, y después la coma decimal (flotante). En general esto se obtiene cambiando el exponente de la base convenientemente.

Luego: f l (x) = 1.011× 22

En el caso general, la expresión de m en binario es: m = (b0.b1 b2 b3...) (2, con b0 = 1

0 E=2 1. 0 1 1 ... 0

! %$# %"$"# 1bit 8bits 23 bits

Ejemplo 2:

x = 71(10 = 1000111 (2

Expresando en forma de número flotante en base B=2, con exponente E=6 ( al desplazar el punto flotante desde la última cifra a la primera), se tiene:

f l (x) = (1.000111) (2 × 26

, siendo su representación en bits:

0 E = 6 1. 000 11 10 ... 0

Notas:

1) El punto decimal que aparece entre b0 y b1 en el tercer campo de 23 bits es mostrado a los

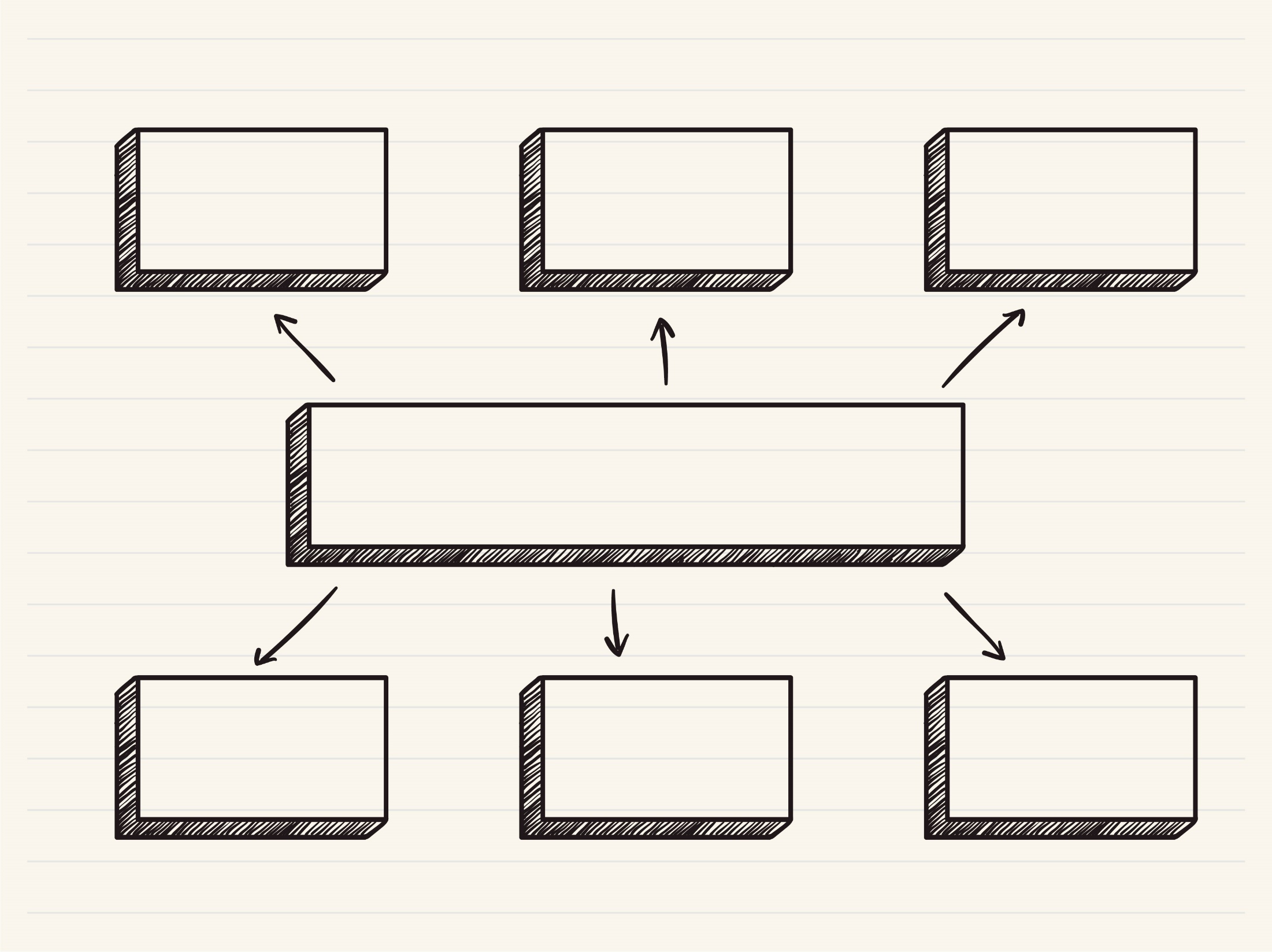
fines didácticos, pero no suele ser almacenado realmente. Por consiguiente, los 23 bits se

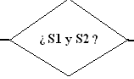
enumeran desde b1 hasta b23.

2) El exponente se coloca en base 10, pero en rigor tiene expresión binaria.

3) El bit del signo: 0 indica un número positivo; 1 es negativo.

Calcula la suma y la resta de los números a = 0.4523 · 104, y b = 0.2115 · 10−3, con una aritmética flotante con mantisa de cuatro dígitos decimales, es decir, una aritmética de cuatro dígitos de precisión. ¿Se produce alguna diferencia cancelaria? Solución. El cálculo es fácil y directo fl(a + b) = 0.4523 · 104 + 0.000 2115 · 100 = 0.4523 · 104 + 0.0000 000 2115 · 104 = 0.4523 · 104 , fl(a − b) = 0.4523 · 104 . Estos cálculos muestran claramente la perdida de dígitos significativos en las operaciones de suma y resta en punto flotante. Observamos que en el caso de la resta no se ha producido una diferencia cancelativa, ya que el resultado tiene una exactitud igual a la precisión (4 dígitos) de la aritmética usada.



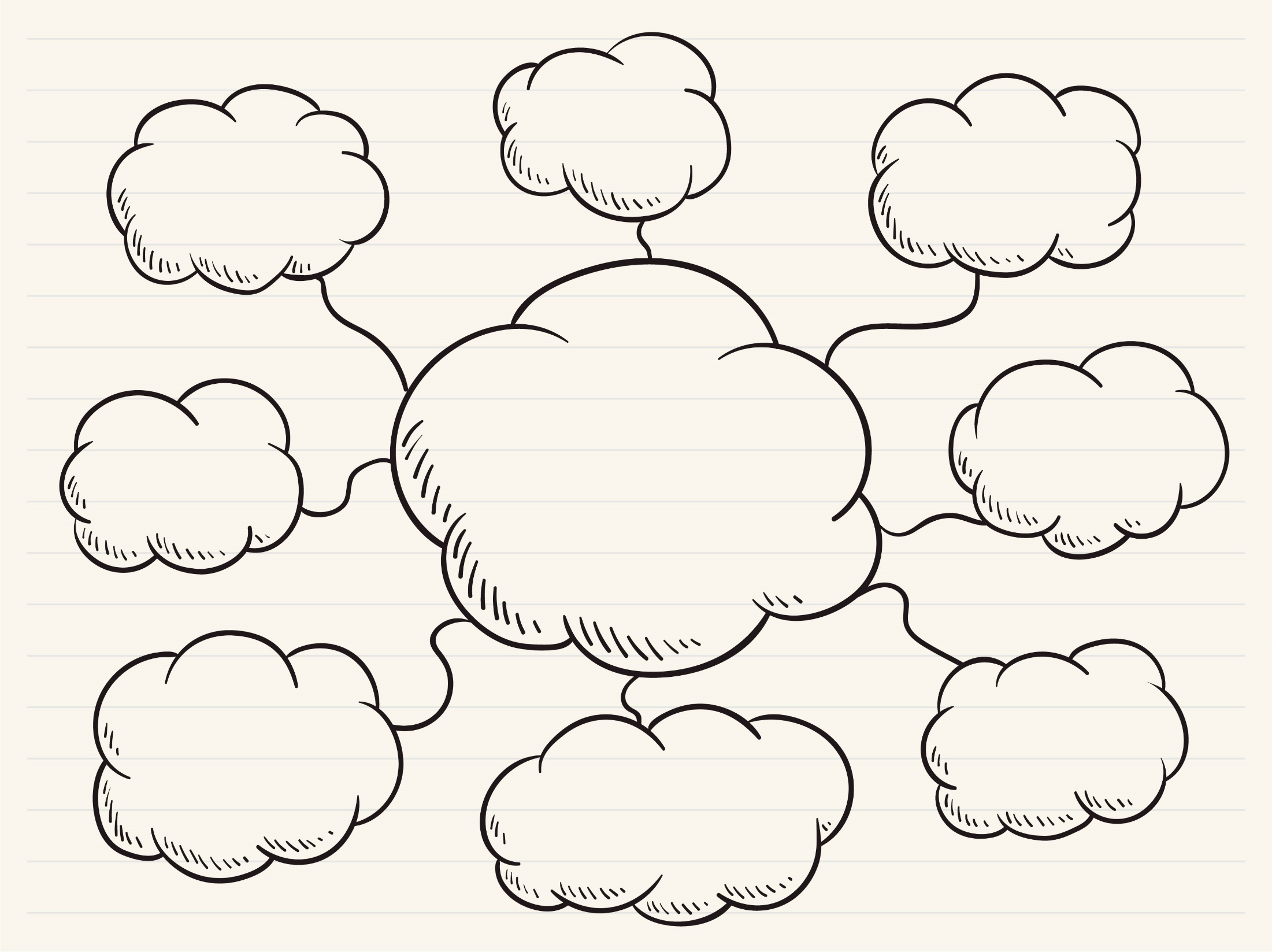
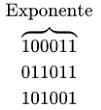
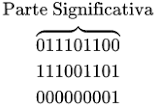
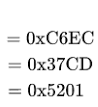




**Aritmetica con enteros**







**Aritmetica con**

**punto flotante**