Tema 7. Aritmética Binaria

- **Objetivos.**
  - Comprender las operaciones aritméticas básicas con los números enteros binarios sin signo.
  - Comprender las características básicas de los sistemas de representación de los números binarios enteros con signo.
  - Asimilar el proceso y las características de las operaciones aritméticas en el sistema de representación en complemento a 2.

Tema 7. Aritmética Binaria

- **Contenido**

1. Operaciones aritméticas con números binarios enteros sin signo.
2. Representación y aritmética de los números binarios enteros con signo.
   2.1. Representación signo-magnitud.
   2.2. Representación en complemento a 2. Operaciones aritméticas.
   2.3. Representación en complemento a 1.
   2.4. Representación en exceso a M.
3. Aritmética BCD.

Operaciones aritméticas con números binarios enteros sin signo

Suma de dos bits.

- La suma de dos dígitos binarios (X_i, Y_i) genera un **bit de suma (S_i)** y otro de **acarreo (C_{i+1})**.
- ***Tabla de la suma de 2 bits.***

X_i +	Y_i	C_{i+1}	S_i
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

- Si la suma es mayor que 1 produce un resultado de 2 bits: bit de suma y de acarreo.
- Se produce acarreo si los dos bits son 1.

- La suma de dos números binarios sigue las mismas reglas que la de los números decimales.
  - Se suman los dos dígitos del mismo peso junto con el acarreo de la suma de los dos dígitos anteriores.
  - Por tanto, hay que usar la tabla de 3 bits: dos dígitos binarios y el acarreo previo.

Operaciones aritméticas con números binarios enteros sin signo

Tabla de suma de 2 dígitos binarios y el acarreo de la etapa anterior.

X_i	$+Y_i$	$+C_i$	C_{i+1}	S_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

- Si la suma es mayor que 1 produce un resultado de 2 bits: bit de suma y de acarreo.
- Se produce acarreo si hay dos o tres bits a 1.
- C_{i+1} y S_i representan en binario adecuadamente el valor de la suma.
- Ejemplos:
  - $1+1+0 = 2 = 10_2 \Rightarrow C_{i+1} = 1, S_i = 0.$
  - $1+1+1 = 3 = 11_2 \Rightarrow C_{i+1} = 1, S_i = 1.$

Operaciones aritméticas con números binarios enteros sin signo

- ***Suma de dos números binarios de n bits.***
  - Empezando por los bits de menor peso se suman los dos bits del mismo peso de cada número junto con el acarreo de la suma de los dos bits anteriores.

Ejemplo 1

$$
\begin{array}{rr cccc l}
 & & 0 & 1 & 1 & 0 & C_i \\
11 & & 1 & 0 & 1 & 1 & X_i \\
+\ \ 3 & + & 0 & 0 & 1 & 1 & Y_i \\
\hline
14 & & 1 & 1 & 1 & 0 & S_i \\
 & & 0 & 0 & 1 & 1 & C_{i+1}
\end{array}
$$

Ejemplo 2

$$
\begin{array}{rr ccccc l}
 & & 1 & 1 & 1 & 1 & 0 & C_i \\
11 & & & 1 & 0 & 1 & 1 & X_i \\
+13 & + & & 1 & 1 & 0 & 1 & Y_i \\
\hline
24 & & 1 & 1 & 0 & 0 & 0 & S_i \\
 & & & 1 & 1 & 1 & 1 & C_{i+1}
\end{array}
$$

  - La suma puede generar un acarreo final (Ejemplo 2).
    - ✓ Indica que el resultado tiene un bit más que los sumandos,
    - ✓ Y que no se puede representar con los n bits del sistema de representación. En los ejemplos n = 4.
    - ✓ Esto indica que se ha producido desbordamiento u overflow.

Operaciones aritméticas con números binarios enteros sin signo

- ***Tabla de la resta de dos dígitos binarios.***

X_i -	Y_i	B_{i+1}	R_i
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

- Genera dos bits para el resultado: el de la resta (R_i) y el del adeudo o préstamo o borrow (B_{i+1}).
- Se produce préstamo si el sustraendo (Y_i) es mayor que el minuendo (X_i).

  - La resta de dos números binarios (X – Y) sigue las mismas reglas que la de los números decimales:
    - Empezando por los bits menos significativos se resta a cada bit del minuendo (X_i) el correspondiente bit del sustraendo (Y_i) y el préstamo generado en la resta del los dos bits anteriores.
    - Por tanto, hay que usar la tabla de 3 bits: dos dígitos binarios y el préstamo anterior.

Operaciones aritméticas con números binarios enteros sin signo.

Resta de números binarios.

- **Tabla de resta de dos dígitos binarios y el préstamo anterior.**

X_i	$-Y_i$	$-B_i$	B_{i+1}	R_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

  - Se opera $X_i - (Y_i + B_i)$.
  - Se produce préstamo si $(Y_i + B_i) > X_i$.
  - Por tanto, si al sumar $(Y_i + B_i)$ se produce acarreo habrá préstamo.

- **Resta de dos números: X – Y**.
  - Primero se comprueba si $X \geq Y$.
  - Si se cumple se hace **X – Y**.
  - Si no se debe realizar la operación inversa **– (Y – X)**.
  - No se produce desbordamiento.

Ejemplo 1: 25 -14

	1	1	1	0	0	B_i
25	1	1	0	0	1	X_i
-14	- 0	1	1	1	0	Y_i
11	0	1	0	1	1	R_i
	0	1	1	1	0	B_{i+1}

Ejemplo 2: 14 – 20 = -(20 – 14) = -6

	1	1	1	0	0	B_i
2 0	1	0	1	0	0	X_i
- 1 4	- 0	1	1	1	0	Y_i
0 6	0	0	1	1	0	R_i $\Rightarrow -00110_2 = -6$
	0	1	1	1	0	B_{i+1}

Operaciones aritméticas con números binarios enteros sin signo

- ***Multiplicación de dos números binarios.***
  - ***Tabla de multiplicación de 2 dígitos binarios.***

X_i	Y_i	P_i
0	0	0
0	1	0
1	0	0
1	1	1

  - Coincide con la multiplicación en decimal.
  - Equivale a la función lógica AND.

- **Multiplicación de X.Y**
  - Se multiplica cada bit del multiplicador por el multiplicando, obteniendo tantos productos parciales como bits tenga el multiplicador.
  - Cada producto parcial será 0, si el bit del multiplicador es 0, o el multiplicando, si el bit del multiplicador es 1.
  - Se suman todos los productos parciales.
  - Suponiendo n y m bits para el multiplicando y el multiplicador, respectivamente, el resultado puede tener n+m bits.

Operaciones aritméticas con números binarios enteros sin signo

- ***División de dos números binarios.***
  - Se opera igual que en decimal.
  - El bit del cociente C_i es 0 si el divisor D es mayor que el dividendo parcial dp_i, y 1 en caso contrario.
  - Si el bit del cociente es 1 se resta el producto de C_i y D, que será D, del dp_i anterior, obteniéndose un nuevo dividendo parcial.
  - La división por 0 es indeterminada.
  - Se genera un resto distinto de 0 si la división no es exacta.

$$
\begin{array}{rr|l}
4 & 1 & 6 \\ \hline
 & 5 & 6
\end{array}
\qquad\qquad
\begin{array}{rrrrrrr|l}
 & 1 & 0 & 1 & 0 & 0 & 1 & 1\ 1\ 0 \\ \hline
- & 1 & 1 & 0 & & & & 1\ 1\ 0\ = 6 \\ \hline
 & 0 & 1 & 0 & 0 & 0 & & \\
- & & & 1 & 1 & 0 & & \\ \hline
 & & 0 & 0 & 1 & 0 & 1 & = 5
\end{array}
$$

Representación de números enteros

Representación en Signo-magnitud

- **Sistema de representación en Signo-magnitud.**
  - Es una modificación del binario natural. Se añade un bit de signo.
  - Considerando un sistema de representación de n bits el bit de mayor peso (n-1) es el bit de signo y los restantes n-1 bits (del 0 al n-2) indican la magnitud del número.
  - **Formato de representación para un sistema de n bits.**

n bits

S	Magnitud (n – 1 bits)

n-1 0

  - ✓ **Magnitud.** Indica en binario natural el valor absoluto del número.
  - ✓ **S es el signo:**
    - ➢ S = 0 Positivo
    - ➢ S = 1 Negativo

Representación de números enteros
Representación en Signo-magnitud

- **Ejemplo. n = 6.**
  - +12 y -12 se diferencian solamente en el bit de signo.

+12	0	0	1	1	0	0
-12	1	0	1	1	0	0

- **Características.**
  - Dos representaciones para el 0.

+0	0	0	0	0	0	0
-0	1	0	0	0	0	0

  - Rango de representación simétrico.
    - Representa la misma cantidad de números binarios enteros positivos que negativos.

Nº positivos	00...00 a 01...11	+0 a $+(2^{n-1}- 1)$
Nº negativos	10...00 a 11...11	-0 a $-(2^{n-1}- 1)$

Rango de representación considerando n = 6

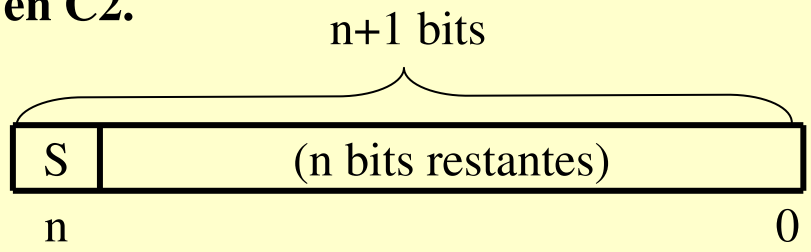
$-(2^5-1)$ a $+(2^5-1) = -31$ a $+31$

Representación de números enteros
Representación en Signo-magnitud

- **Operaciones de suma y resta.** Para sumar y restar números en representación S-M, hay que tener en cuenta la operación y el signo de los operandos:
  - En el caso de la suma, si ambos operandos tienen el mismo signo se suman y el resultado tendrá el mismo signo. Si se trata de operandos de distinto signo se resta al operando de mayor magnitud el operando de menor y se toma como signo el signo del operando de mayor magnitud
  - En el caso de la resta, si ambos operandos tienen el mismo signo, se resta al operando de mayor magnitud el de menor magnitud y se toma el signo del operando de mayor magnitud. Si se tienen operandos de distinto signo se suman y se toma el signo del minuendo.
  - En conclusión hay que realizar tanto operaciones de suma como de resta, además de tener que comparar los operandos.

Representación de números enteros
Representación en complemento a 2

- **Formato de representación de un número A para un sistema de n+1 bits en C2.**



  - ✓ El bit n (S) determina el signo:
    - ➢ S = 0 Nº positivo
    - ➢ S = 1 Nº negativo
  - ✓ Si A es positivo, S=0 y en los n bits restantes se pone el valor de A en binario natural.
  - ✓ Si A es negativo, S0 1 y en los n bits restantes se pone el resultado de la operación 2^n –A . Se puede considerar el bit de signo S como un bit más y hacer la operación 2^{n+1} –A incluyendo en A el bit de signo. Por tanto:
    - ✓ -A = 2^{n+1} –A o lo que es lo mismo, ya que n es un valor genérico y considerando que en A está incluido el signo y se tienen n bits -A = 2^n –A

Representación de números enteros
Representación en complemento a 2

- **Ejemplo. n = 6.**
  - Para representar -12 se hace el C2 a +12, es decir $-12=12_{C2}=2^6-12$.

+12	0	0	1	1	0	0
-12	1	1	0	1	0	0

- **Características.**
  - El 0 tiene una única representación que es 00….00. Se demostrará más fácilmente cuando se estudie el Complemento a 1
    - Rango de representación asimétrico.
      - Al considerarse el 0 positivo, el rango de los nº negativos es una unidad mayor al de los positivos.

Nº positivos	00...00 a 01...11	0 a $+(2^{n-1}-1)$
Nº negativos	10...00 a 11...11	$-(2^{n-1})$ a –1

Rango de representación considerando n = 6

$-(2^5)$ a $+(2^5-1) = -32$ a $+31$

Representación de números enteros
Representación en complemento a 2

- ✓ Para obtener el módulo de un número negativo en C2 se debe aplicar de nuevo la definición de C2 para negativos:
  - ✓ $(A_{C2})_{C2}= (2^n - A)_{C2}= 2^n-(2^n-A)=A$
- ✓ **Resumiendo**, si se tiene un número en C2 se observa el bit más significativo, si es 0, el resto de bits indican la magnitud del número, si es 1, hay que aplicar la definición de complemento a dos para tener la magnitud del número.
- ✓ **Importante: Como se verá más adelante, para hacer el C2 de un número se utilizará el formato Complemento a 1 como formato intermedio.**

Representación y aritmética de los números binarios enteros con signo.
Sistema de representación en complemento a 2. Suma.

- ***Operaciones aritméticas.***
  - ***Suma.***
    - ***Se suman igual que los números binarios incluyendo los bits de signo.***
    - ***Casos:***
      1. ***Suma de dos números del mismo signo. (+X) + (+Y) y (-X) + (-Y)***
         - El resultado es correcto mientras no se produzca desbordamiento.
         - Se produce desbordamiento si se suman dos números del mismo signo y el signo del resultado es distinto al de los operandos.

n = 5 bits

Rango de representación

$-(2^4)$ a $+(2^4-1)$ = -16 a +15

+A = +10 = 01010

+B = +5 = 00101

+C = +13 = 01101

–A = –10 = 10110

–B = –5 = 11011

–C = –13= 10011

a) Suma de dos números positivos (+A) + (+B)

+10	0 1 0 1 0	+A
+ +05	+ 0 0 1 0 1	+B
15	0 1 1 1 1	= +15

$C_f = 0$ $S = 0 \Rightarrow$ nº positivo

- El resultado es positivo, por lo que el bit de signo S = 0
- No se produce acarreo final

Representación y aritmética de los números binarios enteros con signo.

Sistema de representación en complemento a 2. Suma.

n = 5 bits

Rango de representación

$-(2^4)$ a $+(2^4-1)$ = -16 a +15

+A = +10 = 01010

+B = +5 = 00101

+C = +13 = 01101

–A = –10 = 10110

–B = –5 = 11011

–C = –13= 10011

b) Suma de dos números negativos (–A) + (–B)

		1 1 1	
	– 1 0	1 0 1 1 0	–A
+	– 0 5	+ 1 1 0 1 1	–B
	– 1 5	1 1 0 0 0 1	= –15

$C_f = 1$ $S = 1 \Rightarrow$ nº negativo

- El resultado es negativo, por lo que el bit de signo S = 1
- Se produce acarreo final por la suma de los dos bits de signo, que se descarta.

$$(-A) + (-B) = A_{C2} + B_{C2} = 2^n\text{-}A + 2^n\text{-}B = \underbrace{2^n}_{\text{Acarreo final}}+(\underbrace{2^n - A - B}_{\text{Resultado negativo en C2}})$$

c) En la suma de dos números del mismo signo se puede producir desbordamiento.

- Se excede el rango de representación.
- Para representar correctamente el resultado se necesita un bit adicional.

Representación y aritmética de los números binarios enteros con signo.

Sistema de representación en complemento a 2. Desbordamiento.

n = 5 bits

Rango de representación

$-(2^4)$ a $+(2^4-1) = -16$ a $+15$

+A = +10 = 01010

+B = +5 = 00101

+C = +13 = 01101

–A = –10 = 10110

–B = –5 = 11011

–C = –13= 10011

- **Suma de dos números positivos (+A) + (+C)**

		1					
+10		0	1	0	1	0	+A
++13	+	0	1	1	0	1	+C
+23	0	1	0	1	1	1	

$C_f = 0$ $S = 1 \Rightarrow$ nº negativo

- El resultado debe ser positivo, pero se obtiene un bit de signo S = 1, que indica que es negativo.

- **Suma de dos números negativos (–A) + (–C)**

			1	1			
–10		1	0	1	1	0	–A
+–13	+	1	0	0	1	1	–C
–23	1	0	1	0	0	1	

$C_f = 1$ $S = 0 \Rightarrow$ nº positivo

- El resultado debe ser negativo, pero se obtiene un bit de signo S = 0, que indica que es positivo.

- Se produce desbordamiento si se suman dos números del mismo signo y el resultado tiene un signo diferente al de los operandos.
- Para representar el resultado se necesita un bit adicional.

Representación y aritmética de los números binarios enteros con signo.
Sistema de representación en complemento a 2. Suma.

n = 5 bits

Rango de representación

$-(2^4)$ a $+(2^4-1) = -16$ a $+15$

+A = +10 = 01010

+B = +5 = 00101

+C = +13 = 01101

–A = –10 = 10110

–B = –5 = 11011

–C = –13= 10011

2. *Suma de dos números de distinto signo.*

- Consideremos (+X) + (-Y), pero también es aplicable a (-X) + (+Y).
- Se suman en binario sin analizar la magnitud de los operandos.
- El bit de signo indica adecuadamente el resultado.
- Si se produce acarreo final se descarta.
- No se puede producir desbordamiento.
- $(+X) + (-Y) = X + Y_{C2} = X + 2^n - Y$.

a) $|X| \geq |Y| \Rightarrow$ **R = +(X – Y)**

- **Ej. (+A) + (-B) y (+A) + (-A)**

$$
\begin{array}{rr r l}
 & 1\ \ \ \ 1\ \ \ \ & & \\
+10 & 0\ 1\ 0\ 1\ 0 & +A \\
+-05 & +\ 1\ 1\ 0\ 1\ 1 & -B \\
\hline
+05 & 1\ 0\ 0\ 1\ 0\ 1 &
\end{array}
\qquad
\begin{array}{rr r l}
 & 1\ 1\ 1\ \ \ \ & & \\
+10 & 0\ 1\ 0\ 1\ 0 & +A \\
+-05 & +\ 1\ 0\ 1\ 1\ 0 & -A \\
\hline
+05 & 1\ 0\ 0\ 0\ 0\ 0 &
\end{array}
$$

$C_f = 1$ $\quad S = 0 \Rightarrow$ nº positivo $\qquad C_f = 1$ $\quad S = 0 \Rightarrow$ nº positivo

- El resultado es positivo.
- Se produce acarreo que se descarta.

- **Efectivamente:** $\mathbf{X + 2^n - Y = 2^n + (X - Y) = 2^n + R \geq 0}$

Representación de números enteros
Representación en complemento a 1

- **Formato de representación de un número A para un sistema de n+1 bits en C1.**

n+1 bits

S	(n bits restantes)
n	0

✓ El bit n (S) determina el signo:
   - ➢ S = 0 Nº positivo
   - ➢ S = 1 Nº negativo

✓ Si A es positivo, S=0 y en los n bits restantes se pone el valor de A en binario natural.

✓ Si A es negativo, S0 1 y en los n bits restantes se pone el resultado de la operación $2^n - A - 1$. Se puede considerar el bit de signo S como un bit más y hacer la operación $2^{n+1} - A - 1$ incluyendo en A el bit de signo. Por tanto:

   ✓ $-A = 2^{n+1} - A - 1$ o lo que es lo mismo, ya que n es un valor genérico y considerando que en A está incluido el signo y se tienen n bits $-A = 2^n - A - 1$

Representación de números enteros
Representación en complemento a 1

- **Sistema de representación en complemento a 1.**
  - El formato es igual que el de C2, excepto que los números negativos se representan haciendo el C1 del número positivo.
  - $-A = 2^n - 1 - A$
  - Al hacer el C1 se obtiene un bit de signo S = 1.
  - Se tiene la siguiente relación entre C1 y C2:
    - **C2=C1+1**

Representación de números enteros

Representación en complemento a 1

- **Ejemplo. n = 6.**
  - Para representar -12 se hace el C1 a +12.

+12	0	0	1	1	0	0
-12	1	1	0	0	1	1

  - **Como se puede observar en la representación C1 para representar un número negativo basta con invertir todos los bits del número en su representación binario natural**

C2=C1+1 : Por tanto para convertir un número a C2 se invierten los bits del número (incluido el bit de signo) y se le suma 1

- **Características.**
  - El 0 tiene dos representaciones.

+0	0	0	0	0	0	0
-0	1	1	1	1	1	1

  - Rango de representación simétrico.
    - Al tener el 0 doble representación, el rango de los nº positivos y negativos es igual.

Nº positivos	00...00 a 01...11	$+0$ a $+(2^{n-1} - 1)$
Nº negativos	11...11 a 10...00	-0 a $-(2^{n-1} - 1)$

Rango de representación considerando n = 6

$-(2^5 - 1)$ a $+(2^5 - 1) = -31$ a $+31$

Representación de números enteros

Representación en complemento a 1. Suma

- En el caso de la operación de suma en complemento a 1 sigue unas reglas similares a C2, sin embargo, en C1 es necesario hacer unas correcciones después de hacer la suma. Como no se va utilizar representación de C1 para realizar sumas, sino como formato intermedio en esta asignatura, vamos a mostrar un ejemplo de la necesidad de estas correcciones.
- **Ejemplo suma de -13+21. El resultado debe ser 8**
  - Supongamos 6 bits para la representación.
  - -13, como es negativo su representación se obtiene a partir de 13(001101) invirtiendo 110010.

$$
\begin{array}{r} -13 \\ +23 \\ \hline +8 \end{array}
\qquad
\begin{array}{rcccccc} & 1 & 1 & 0 & 0 & 1 & 0 \\ + & 0 & 1 & 0 & 1 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{array}
$$

acarreo

El resultado es 7. La corrección consiste en sumar 1 si hay acarreo. Entonces se obtiene el valor correcto

Representación de números enteros

Representación del cero en complemento a 2. Demostración utilizando C1

- ***Representación del cero en C2***
- El cero tiene una única representación en c2:
  - Si se considera la definición de números positivos para el valor cero entonces:
    - ✓ En el signo se pone 0 y en el resto se pone la magnitud

0	0	0	…..	0	0

    - ✓ Si se considera la definición de número negativo entonces C2=C1+1. Para tener el complemento a 1 se invierten todos los bits con lo que se tiene 11……..11 y al sumar 1 y despreciar el acarreo final se tiene 00…….00. Por tanto coincide con lo anterior, es decir, el cero en C2 tiene una única representación que es 000…0000

Representación y aritmética de los números binarios enteros con signo.

Sistema de representación en exceso a M.

- **Sistema de representación en exceso a M.**
  - Esta representación se puede utilizar para representar números enteros con signo. La representación en exceso M se mostró en el tema 2 (Representación de la Información) y no se va a utilizar en esta asignatura para realizar operaciones aritméticas por lo que no se va a profundizar mucho más. Se verá esta representación en otras asignaturas con más profundidad

Aritmética BCD. Suma.

- ***Suma de dos números BCD 8421.***
  - El código BCD 8421 es un código binario ponderado de 4 bits, que representa los 10 dígitos decimales. Por tanto,
    - Usa solamente las combinaciones del 0 (0000) al 9 (1001).
    - No usa las combinaciones del 10 (1010) al 15 (1111).
  - Como las combinaciones del código BCD coinciden con las del sistema de representación binario:
    - Los dígitos BCD de mismo peso de ambos números se suman en binario.
    - Si el resultado de la suma es mayor de 9, entonces se debe hacer una corrección sumando 6 al resultado.

Si se considera el resultado como dígitos BCD el resultado obtenido es 12 (1)(0010) en lugar de 18 (1)(1000)

$$
\begin{array}{r} \\ 9 \\ +9 \\ \hline 18 \end{array}
\qquad
\begin{array}{rrrrrr}
 & 1 & & & & \\
 & 1 & 0 & 0 & 1 & \\
+ & 1 & 0 & 0 & 1 & \\ \hline
1 & 0 & 0 & 1 & 0 & = 18
\end{array}
$$

Resultado de la suma binaria

$$
\begin{array}{r} \\ 9 \\ +9 \\ \hline 18 \end{array}
\qquad
\begin{array}{rrrrrrr}
 & & 1 & & & & \\
 & & 1 & 0 & 0 & 1 & \\
 & + & 1 & 0 & 0 & 1 & \\ \cline{2-6}
 & 1 & 0 & 0 & 1 & 0 & = 18 \\
 & & + & 0 & 1 & 1 & 0 \\ \cline{2-6}
000 & 1 & 1 & 0 & 0 & 0 &
\end{array}
$$

Resultado corregido sumando 6

Bibliografía detallada
Aritmética Básica

Las diapositivas se han confeccionado utilizando como fuente:

- Aritmética Básica

  "Diseño Lógico". A. Lloris, A. Prieto. Mc-Graw Hill. 1996. Capítulos: 4.2.1, 4.2.2, 4.2.3, 4.2.4, 4.2.5