TRABAJO PRÁCTICO Nº 1 - EJERCICIOS RESUELTOS

1) a) Convertir a binario: 124_{10}

```
124 |_2_
 04   62 |_2_
  0    02  31 |_2_
        0  11  15 |_2_
            1   1   7 |_2_
                    1   3 |_2_
                        1   1
```

Rta.: $124_{10} = 1111100_2$

b) Idem a octal:

```
124 |_8_
 44  15 |_8_
  4   7   1
```

Rta.: $124_{10} = 174_8$

c) Idem a hexadecimal:

```
124 |_16_
 12   7
```

El resto 12 en hexadecimal se representa con la letra C, por lo tanto $124_{10} = 7C_{16}$

2) Hacer las conversiones:

a) $\mathbf{717_8}$ a decimal:

$717_8 = \mathbf{7} \times 8^2 + \mathbf{1} \times 8^1 + \mathbf{7} \times 8^0 = 448 + 8 + 7 = 463$

b) A3CBEFD $_{\mathbf{16}}$ a binario

$$\underset{1010}{\underbrace{A}}\ \underset{0011}{\underbrace{3}}\ \underset{1100}{\underbrace{C}}\ \underset{1011.}{\underbrace{B}}\ \underset{1110}{\underbrace{E}}\ \underset{1111}{\underbrace{F}}\ \underset{1101}{\underbrace{D}}\ {}_{16 \to 2}$$

$A3CBEFD_{16} \to 1010\ 0011\ 1100\ 1011\ \ 1110\ 1111\ 1101_2$

c) 1F4 $_{\mathbf{16}}$ a octal:

$$\begin{array}{cccc} 1 & F & 4 & {}_{16} \\ 000\ 1 & 11\ 11 & 0\ 100 & {}_2 \end{array}$$

$$\underbrace{000}_{0}\ \underbrace{111}_{7}\ \underbrace{110}_{6}\ \underbrace{100}_{4}\ {}_8$$

$1F4_{16} \to 764_8$

d) $\mathbf{4275_8}$ a hexadecimal:

$$\begin{array}{cccc} 4 & 2 & 7 & 5 & {}_8 \\ 100 & 0\ 10 & 11\ 1 & 101 & {}_2 \end{array}$$

$$\underbrace{1000}_{8}\ \underbrace{1011}_{B}\ \underbrace{1101}_{D}\ {}_{16}$$

$4275_8 \rightarrow 8BD_{16}$

3) Efectuar la siguiente resta (usando módulo 8 o 16, según sea necesario):

a) Usando complemento a dos

101001_2 - 1101_2 = 101001 + (- 1101)

Calculo el complemento a la base 2 del (- 1101) y luego sumo:

```
                   100000000
                 -  00001101
                    11110011
                 +  00101001
Se descarta   → 1| 00011100
```

Nota: una forma directa de calcular el Ca2 es copiar los bits desde el menos significativo hasta el primer uno y luego invertir los restantes.

Resultado: **00011100**

5043_8 - 100011001_2 = 101 000 100 011 + (- 100011001)

En este caso, necesito usar un formato de 16 bits.
El Ca2 de (- 100011001) es: 1111111011100111.
Realizo la suma en 16 bits:

```
                0000101000100011
             +  1111111011100111
Se descarta → 1| 0000100100001010
```

Resultado (en 16 bits): **0000100100001010**

b) Usando complemento a uno

1001101_2 - 11111_2 = 1001101 + (-11111)

Calculo el complemento a uno de (-11111) en 8 bits.

```
   100000000
 -         1
    11111111
 -  00011111
    11100000
```

Nota: también puedo hallar el complemento a uno invirtiendo los 8 bits del sustraendo 00011111

Ahora hago la suma en 8 bits:

```
                     01001101
                   + 11100000
Hay que sumarlo  → 1| 00101101
                   +        1
                     00101110
```

Como la suma (de un número positivo y uno negativo) dio un resultado positivo, hay que sumar 1 para hacer la corrección debida a la doble representación del cero. Este problema no

se presenta en las operaciones en complemento a 2 ya que la representación del cero es única.
Resultado (en 8 bits): **00101110**

4) a) Indicar la notación exceso 127 normalizado de:

N = -123,7605$_8$

$N = \pm\ m \times 10^{\pm p} = \pm 1{,}f \times 10^{\pm p}$

$N = -123{,}7605_8 = -\ 001\ 010\ 011,\ 111\ 110\ 000\ 101$
$= -\ 1{,}010011111110000101 \times 10^{110}$

Como el exponente es positivo, le sumo 127:

$$\begin{array}{r} 01111111 \\ +\ \underline{00000110} \\ 10000101 \end{array}$$

Para la representación se emplean 4 bytes (32 bits):

1 1000010	1 0100111	11110000	10100000

signo característica mantisa

N = -1101
$N = -1{,}101 \times 10^{011}$
Como el exponente es positivo (011), le sumo 127:

$$\begin{array}{r} 01111111 \\ +\ \underline{00000011} \\ 10000010 \end{array}$$

N = -1101 **:**

11000001	01010000	00000000	00000000

b) ¿A qué números hexadecimales representan las siguientes notaciones exceso 127?

I)

01111011	10000011	00111110	00000000

$$\begin{array}{rl} 11110111 & \text{exponente} \\ -\ \underline{01111111} & 127_{10} \\ 01111000 & 120_{10} \end{array}$$

$N = +\ 1{,}00000110011111 \times 10^{01111000}$

Para pasar a hexadecimal, debo tener en cuenta que la posición de la coma se modifica ya que la cantidad de dígitos es 4 veces menor (cada dígito hexadecimal equivale a 4 dígitos binarios). Por lo tanto el exponente 01111000 = 120_{10} pasa a ser 30_{10}, es decir 1E en hexadecimal. En definitiva nos queda:

N = + 1,067C x 10^{1E}

5) Determinar los equivalentes hexadecimales:

a) **92 D9 BC F0 =** 1001 0010 1101 1001 1011 1100 1111 0000

Exponente: 01111111 - 00100101 = 01011010 = -90_{10}

Su representación en punto flotante simple precisión es:

$$N = -1{,}1011001101111001111 \times 10^{\overbrace{-01011010}^{-90_{10}}} = -0110{,}1100\ 1101\ 1110\ 0111\ 1000 \times 10^{\overbrace{-01011100}^{-92_{10}}}$$

Hago la conversión tomando de a cuatro bits y reemplazando por su equivalente en hexadecimal. Por lo tanto, la posición de la coma se corre un número tal que resulta de dividir al exponente (-92_{10}) también por 4, quedando (-23_{10}), que en hexadecimal es (- 17). Así que en definitiva queda:

$$\mathbf{N} = -6{,}CDE78 \times 10_h^{-23\ d} = \mathbf{-6{,}CDE78 \times 10_h^{-17\ h}}$$

6) Representar en formato BCD:

Simplemente se representa cada dígito decimal (0 a 9) con 4 bits:

4099_{10} = 4 0 9 9 = 0100 0000 1001 1001