TP info 12 - Corrigé

I1. Base deux

1. $11010001 = 1\times2^0 + 0\times2^1 + 0\times2^2 + 0\times2^3 + 1\times2^4 + 0\times2^5 + 1\times2^6 + 1\times2^7$

$= 1 + 16 + 64 + 128$

$= 209$

2. 766 en base 2 ?

766	2
0	383
1	191
1	95
1	47
1	23
1	11
1	5
1	2
0	1
1	0

$766 = 1011111110$

I2. Base hexadécimale

1. $a = 10$, $b = 11$, $c = 12$, $d = 13$, $e = 14$, $f = 15$

2. $(ef54a)_{16} = 10\times16^0 + 4\times16 + 5\times16^2 + 15\times16^3 + 14\times16^4$

$= 980\,298$

3.

15671	16
7	979
3	61
13 = d	3
3	0

$15671 = (3d37)_{16}$

I3. Complément à deux

1. $n = 8$ bits : $-2^{n-1} \leq N \leq 2^{n-1} - 1$

ici $-2^7 \leq N \leq 2^7 - 1$

$-128 \leq N \leq 127$

2. Divisions successives par 2 :

101 | 2 → 50, reste 1
50 | 2 → 25, reste 0
25 | 2 → 12, reste 1
12 | 2 → 6, reste 0
6 | 2 → 3, reste 0
3 | 2 → 1, reste 1
1 | 2 → 0, reste 1

101 = 01100101   ⚠ 8 bits !

−101 ?

```
   01100101
      ↓
   10011010
puis
 +        1
 ----------
   10011011   = −101
```

Somme

```
  101            01100101
− 101    ⟷     + 10011011
                ----------
                X00000000
```

↑ bit excédentaire.

99 = 01100011
57 = 00111001

```
   99            01100011
 + 57          + 00111001
 ----          ----------
  156            10011100
```

156 : valeur attendue.

10011100 ↑ négatif : absurde !

Explication : 156 n'est pas représentable en complément à 2 sur 8 bits !

Dépassement de capacité.

I4 - Conversion binaire → décimale.

```
def bin_to_dec(a):
    """
    N : list
    Bits de poids fort à gauche
    """
    x=0
    for i in range(len(a)):
        x=x+a[i]*2**(len(a)-i-1)
    return x
```

$a = a_n a_{n-1} \cdots a_1 a_0$

$x = \sum_{i=0}^{n} a_i \cdot 2^i$

Il faut

I5 - Conversion décimal -> binaire

```
def dec_to_bin(n):
    b=[]
    while n>0:
        b.append(n%2)
        n=n//2
    b.reverse()
    return b

print(dec_to_bin(10))
```

15 | 2 → 1, 7 | 2 → 1, 3 | 2 → 1, 1 | 2 → 1, 0

le quotient devient dividende

condition d'arrêt. le quotient vaut 0

renvoie [1,0,1,0].

I6. Addition binaire

```
def add_bin(a,b,N):
    """
    Calcule c=a+b en representation binaire
    a,b,c : list, liste de taille N. Representation binaire
    en complement a deux. Bits de poids fort a l'indice 0.
    N : taille de l'entier en bits.
    """
    c=init_list(N)#liste nulle de longueur N
    i=N-1#initialisation au bit de poids faible
    r=0#retenue
    while i>=0:
        s_bit=a[i]+b[i]+r
        if s_bit==0:
            c[i]=0
            r=0
        elif s_bit==1:
            c[i]=1
            r=0
        elif s_bit==2:
            c[i]=0
            r=1
        else:#a[i]+b[i]+r==2
            c[i]=1
            r=1
        i=i-1
    return c

print(add_bin([0,1,1,1],[0,0,0,1],4))#-1+1 en complement a deux
```

Renvoie [0,0,0,0] : fonctionne !

```
def init_list(N):
    l=[]
    for i in range(N):
        l.append(0)
    return l
```