La conversion de binaire à décimal pouvant être sérieusement rébarbative, elle semble être une tâche idéale pour un ordinateur. Dans ce TP, nous allons donc travailler sur les codes permettant la conversion d'un nombre binaire en représentation décimale.

Pour vous aider, les squelettes des codes sont donnés à chaque fois. Votre travail consiste à

- Compléter le code fournis et ajouter les commentaires nécessaires
- Tester le code

Dans chaque cas, les nombre binaires sont représentés par une liste ou chaque élément correspond à 1 bit. Les numéros de ligne donnés sont ceux du fichier python fournis.

POUR CHAQUE FONCTION SUR LAQUELLE VOUS TRAVAILLEZ, DECOMMENTER LE SQUELETTE CORRESPONDANT.

Binaire à décimal

La fonction `BtoD` prend en argument une liste `binaire` correspondant au nombre binaire à convertir, et renvoi un entier `res`, le résultat de la conversion.

Note : Il faut bien étudier la relation entre les indices des cases et les puissances de 2 correspondantes.

```
2  def BtoD(binaire):
3      N = ..... # nombre de bits du nombre
4      .....
5      #on lit les bits
6      for .....
7          poids = ..... #poids (valeur) du bit lu
8          .....
9      .....
```

Test possible : assert BtoD([1,0,1,1]) == 11

Entiers relatifs par la méthode du complément à 2

La fonction `BtoD_Relatif8` prend en argument une liste `binaire` correspondant au nombre binaire sur 1 octet à convertir, et renvoi un entier `res`, le résultat de la conversion, en utilisant la fonction `BtoD`.

```
13 def BtoD_Relatif8(binaire):
14     #vérification qu'on est bien sur 1 octet
15     if .....:
16         print("sur plus d'un octet")
17         return
18     .....
19     .....
20     .....
21     .....
22     .....
```

Test possible :

assert BtoD_Relatif8([0,0,0,0,1,1,0,1]) == 13 and BtoD_Relatif8([1,1,1,0,1,1,0,1]) == -19

Binaire flottant à décimal

La fonction `BFLOTtoD` prend en argument une liste `binaire` correspondant au nombre binaire à virgule à convertir, en incluant le point de séparation entre guillemets, et renvoi un flottant `res`, le résultat de la conversion, en utilisant la fonction `BtoD`.

Note : Il faut, là aussi, bien étudier la relation entre les indices des cases et les puissances de 2 correspondantes.

```
26 def BFLOTtoD(binaire):
27     # on sépare les unités des flottants en créant 2 listes
28     unit = []
29     flott = []
30     virgule = 0
31     for elem in binaire:
32         if elem == ".":
33             virgule = 1
34         elif virgule == 0:
35             unit = .....
36         else:
37             flott = .....
38     res = .....# conversion des unités en décimal
39     N=..... # nombre de bits après la virgule
40     #on lit les bits
41     for .....
42         poids = ..... #poids (valeur) du bit lu
43         res = .....
44     .....
```

Test possible : assert BFLOTtoD([1,0,1,1,".",0,1]) == 11.25

Flottant sur 32 bits

La fonction BtoD_32bits prend en argument liste mot correspondant au mot binaire à convertir, et renvoi le résultat de la conversion, en utilisant les fonctions BtoD_Relatif8 et BFLOTtoD.

```
48 def BtoD_32bits(mot):
49     # On sépare les différentes parties du nombre
50     signe = .....  #1 bit pour le signe
51     exposant = .....  #8 bits pour l'exposant
52     mantisse = .....  #23 bits pour la mantisse
53
54     .....
55     .....
56     .....
57     .....
58     .....
59     .....
```

Test possible :

assert BtoD_32bits([0,0,0,0,0,0,0,1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]) == 9.5

Test sur les flottants

Ecrire la commandes suivante dans la console : 0.1+0.2 == 0.3
Que remarque-t-on ?

Une fonction DFLOTtoB(Nb) est fournis. Elle converti un décimal flottant en binaire avec 23 bits flottants (il n'est pas nécessaire de comprendre ce code) et, dans la console, écrire les commandes suivantes :

- DFLOTtoB(0.1)
- BFLOTtoD(DFLOTtoB(0.1))

On remarque que BFLOTtoD(DFLOTtoB(0.1)) donne 0.09999990463256836.
Expliquer alors le résultat de 0.1+0.2 == 0.3

Pour aller un peu plus loin…

Conversion hexadécimal-décimal : Ecrire une fonction HtoD prend en argument une liste hexadec correspondant au nombre hexadécimal à convertir (et pouvant donc contenir des caractères entre guillemets), et renvoi le résultat de la conversion en décimal.