DM : Nombres négatifs et complément à 2

Le but est de compléter le programme écrit durant le TP noté et d'ajouter des fonctions permettant de convertir des nombres négatifs de la base 10 à la base 2 en utilisant la méthode du complément à 2. *On n'utilisera que les fonctions vues en cours.*

1) **a.** Par rapport aux notes que vous avez prises en cours, rappelez les différentes étapes nécessaires pour convertir un nombre négatif de la base 10 à la base 2.
   b. En déduire les différentes fonctions qui seront nécessaires pour effectuer cette conversion et expliquer le but de chaque fonction.

2) Créez un fonction `osBits` qui ajoute des bits manquants si un nombre binaire n'est pas codé sur 8 ou 16 bits ou enlève des bits en trop si le nombre binaire est codé sur plus de 8 ou 16 bits.
   Entrée : un nombre binaire **n**, un nombre entier voulu de bits **`maxBits`**
   Sortie : un nombre binaire écrit avec **`maxBits`** bits.
   Astuce : Aidez-vous de la fonction `fillBits` écrite dans le programme convertisseur.py, disponible sur Pronote. On pourra utiliser l'instruction `tab[a:b]` pour sélectionner les éléments de la liste tab compris entre a et b-1.

3) La fonction addBin vise à additionner deux nombres binaires a et b.
   a. Compléter cette fonction à l'aide des commentaires et de la fonction `osBits`.

```
1  def addBin(a:int ,b:int ) -> int:
2      a =                                  # passe a sur 8 bits et transforme en string.
3      b =                                  # passe b sur 8 bits et transforme en string.
4      c = ''
5      rem = 0
6      for i in range(7, -1, -1):
7          S =                                          # additionne le i-ème élément
   des strings a et b, convertit en entiers, avec l'entier rem.
8          if S < 2 :
9              c = str(S) + c
10             rem = 0
11         elif S == 2 :
12             c = '0'+ c
13             rem = 1
14         elif S == 3 :
15             c = '1'+ c
16             rem = 1
17     return                                   # renvoie un entier c sur 8 bits
```

b. Quelles sont les valeurs prises par la variable i dans la boucle définie ligne 6 ?

c. Que représentent les variables S et `rem` ?

d. Expliquer le fonctionnement de cette algorithme.

4) À l'aide des questions précédentes et en utilisant la fonction dec2bin du programme convertisseur.py, écrire une fonction dec2signedBin qui convertit un nombre positif ou négatif en base 10 en binaire signé sur 8 bits.
Entrée : un entier relatif en base 10
Sortie : un entier relatif en base 2
Astuce : on pourra créer deux chaines de caractères.

Début de la fonction :

```
1  def dec2signedBin(n:int) -> int :
2      if n>= 0:
3      else:
4
5
6
7      return
```