Informatique PCSI
TP 11b : représentation des entiers

Exercice 1

Liste des chiffres

1. Écrire une fonction qui prend en paramètre un entier naturel et renvoie la liste des chiffres de son écriture binaire rangés dans l'ordre inverse. Pour $n = 11$, on obtient la liste `[1, 1, 0, 1]`
2. Écrire une fonction qui prend en paramètre la liste des chiffres de l'écriture binaire d'un entier naturel rangés dans l'ordre inverse et renvoie cet entier naturel.

Exercice 2

Écrire une fonction `ajoute` qui prend en paramètres deux nombres `n` et `m`, la base utilisée `b` et un entier `i` et renvoie le nombre `n + m * b**i`. Chaque nombre est représenté par la liste inversée des chiffres de son écriture en base `b` (comme dans l'exercice précédent).

Pour tester la fonction, écrire à la suite de la définition :

```
a = [1 ,1 ,0 ,1]
b = [1, 0, 0, 1]
assert ajoute([0], b, 2, 0) == b
assert ajoute(a, [1], 2, 0) == [0, 0, 1, 1]
assert ajoute(a, b, 2, 0) == [0, 0, 1, 0, 1]
```

Poser l'opération pour comprendre les différents mécanismes. Il faut penser à traiter tous les cas, avec les retenues. Tester la fonction en cours d'écriture.

Exercice 3

Chaque nombre est représenté comme dans les deux exercices précédents.

1. Écrire une fonction `produit1` qui prend en paramètres deux nombres, dont le second s'écrit avec un seul chiffre, et la base utilisée et renvoie le produit des deux nombres.

   Pour tester la fonction, écrire à la suite de la définition :

```
    assert produit1([1, 0, 1, 1], [0], 2) == [0] # en base 2
    assert produit1([1, 0, 1, 1], [1], 2) == [1, 0, 1, 1]
    assert produit1([2, 3, 4], [2], 10) == [4, 6, 8] # en base 10
    assert produit1([3, 4, 5], [3], 10) == [9, 2, 6, 1]
    assert produit1([2, 3, 4], [5], 16) == [10, 15, 4, 1] # en base 16
    assert produit1([2, 3, 4], [6], 16) == [12, 2, 9, 1]
```

2. Écrire une fonction `produit` qui prend en paramètres deux nombres et la base utilisée et renvoie le produit des deux nombres. Utiliser les fonctions `ajoute` de l'exercice précédent et `produit1`.

   Pour tester la fonction, écrire à la suite de la définition :

```
assert produit([1, 0, 1], [1, 1], 2) == [1, 1, 1, 1]
assert produit([1, 1, 1], [1, 1], 2) == [1, 0, 1, 0, 1]
assert produit([7, 2, 8], [5, 2], 10) == [5, 7, 6, 0, 2]
assert produit([4, 1], [4], 16) == [0, 5]
```