Informatique PCSI
TP 11b : représentation des entiers

Exercice 1

1.

```
def chiffres(n):
    if n == 0:
        return [0]
    rep = []
    while n != 0:
        rep.append(n % 2)
        n = n // 2
    return rep
```

2. Une simple boucle permet d'accéder à chaque chiffre.

```
def entier(chiffres):
    n = 0
    for i in range(len(chiffres)):
        n = n + chiffres[i] * 2**i
    return n
```

Exercice 2

```
def ajoute(n, m, b, i): # somme de n + m * b**i
    rep = list(n) # rep contient la réponse
    if m == [0]:
        return rep
    for k in range(len(m) + i - len(n)):
        rep.append(0) # si m * b**i plus long que n
    retenue = 0
    for k in range(len(m)): # sommes des chiffres n[i+k] + m[k]
        s = rep[i] + m[k] + retenue
        if s >= b: # traitement de la retenue
            rep[i] = s - b
            retenue = 1
        else:
            rep[i] = s
            retenue = 0
        i = i + 1
    while i < len(n) and retenue == 1: # si n plus long que m * b**i
        s = n[i] + retenue
        if s >= b:
```

```
            rep[i] = s - b
            retenue = 1
        else:
            rep[i] = s
            retenue = 0
        i = i + 1
    if retenue > 0: # s'il reste une retenue finale
        rep.append(retenue)
    return rep

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]
```

Exercice 3

1.

```
def produit1(n, c, b):
    if c == [0]:
        return [0]
    if c == [1]:
        return n
    # en base deux, c'est terminé
    retenue = 0
    prod = len(n) * [0]
    for i in range(len(n)):
        p = n[i] * c[0] + retenue
        prod[i] = p % b
        retenue = p // b
    if retenue > 0:
        prod.append(retenue)
    return prod

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.

```
def produit(n, m, b):
    prod = []
    for i in range(len(m)):
```

```
        p = produit1(n, [m[i]], b) # [m[i]] est un nombre à un chiffre
        prod = ajoute(prod, p, b, i)
    return prod

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]
```