Récursivité et arithmétique

« La Mathématique est la reine des sciences et l'Arithmétique est la reine des mathématiques. »

Définition : division euclidienne dans $\mathbb{N}$

Soient A et B deux entiers naturels, et $B \neq 0$. Il existe deux nombres uniques Q et R (vérifiant $0 \leqslant R < B$) tels que l'on puisse écrire

$$A = Q \times B + R$$

C'est exactement la division que l'on a apprise à l'école primaire (celle où l'on s'arrête aux nombres entiers) :

$$\begin{array}{c|c} A & B \\ \hline R & Q \end{array}$$

- A est appelé le *dividende* ;
- B est le *diviseur* ;
- Q est le *quotient* ;
- R est le *reste*, il est *impérativement* plus petit que B.

En PYTHON on obtient Q en évaluant `A // B` et R en évaluant `A % B`, cette dernière opération se lit « A modulo B ». Voici un exemple

Shell Python

```
>>> 22 // 7
3
>>> 22 % 7
1
```

Exercice 1

1. En PYTHON, écrire une fonction (non-récursive) `units_digit` qui
   - en entrée prend un `int` positif ;
   - renvoie un `int` qui est son chiffre des unités.

2. De même écrire une fonction (non-récursive) `hundreds_digit` pour le chiffre des centaines.

3. De même pour une fonction (non-récursive) `thousands_digit` qui renvoie le chiffre des milliers.

Exercice 2

Écrire une fonction récursive `decimal_length` basée sur // et/ou % qui

- en entrée prend un `int` positif;
- renvoie un `int` qui est le nombre de chiffres de l'écriture décimale de ce nombre.

Exercice 3

En s'inspirant de l'exercice précédent : Écrire une fonction récursive `binary_length` basée sur // et/ou % qui

- en entrée prend un `int` positif;
- renvoie un `int` qui est le nombre de chiffres de l'écriture binaire de ce nombre.

Exercice 4

On considère le procédé suivant :

- soit $m \in \mathbb{N}$ un entier écrit en écriture décimale $m = (a_p \cdots a_1 a_0)_{10}$,
  par exemple $m = 31\,976$;

- on « coupe » cette écriture en deux au niveau des unités : avec m on forme $m_1 = (a_p \cdots a_1)_{10}$ et $m_2 = (a_0)_{10}$,
  pour notre exemple $m_1 = 3\,197$ et $m_2 = 6$;

- On calcule $m' = m_1 - 2m_2$,
  pour notre exemple cela donne $m' = 3\,197 - 2 \times 6 = 3185$

1. à l'aide de // et/ou % écrire une fonction `f` qui

- en entrée prend un `int` positif `m` ;
- en sortie renvoie `m'`.

2. En fait, la fonction f donne un critère de divisibilité par 7 pour un entier m :

- si $m \leqslant 70$ alors s'il appartient à -7 ; 0 ; 7 ; 14 ; 21 ; 28 ; 35 ; 42 ; 49 ; 56 ; 63 ; 70, m est divisible par 7, sinon il ne l'est pas ;
- sinon on regarde si $f(m)$ est divisible par 7.

Pour notre exemple on obtient

$$31\,976 \mapsto 3\,197 - 2 \times 6 = 3\,185 \mapsto 318 - 2 \times 5 = 308 \mapsto 30 - 2 \times 8 = 14$$

et on en conclut qu'il est divisible par 7.

Programmer une fonction récursive `is_divisible_by_7` qui

- en entrée prend un `int` positif ;
- en sortie renvoie `True` ou `False` selon que l'entier est divisible par 7.

Cette fonction utilisera la fonction `f` définie précédemment.