Travaux Dirigés de Programmation Impérative n°1

Cours d'informatique de Première Année

—Licence MIEF/SPI—

Récursivité et Pile d'appel

La récursivité est une méthode de programmation très puissante. Elle permet d'écrire des programmes faciles à comprendre et souvent efficaces bien que son principal inconvénient est d'obliger le compilateur à utiliser une pile pour mémoriser les calculs intermédiaires.

▸ Exercice 1. Factorielle

1. *La* `factorielle` *d'un entier positif ou nul* n*, notée* $n!$*, est définie par*

$$n! = \begin{cases} 1 & \text{si } n = 0, \\ (n-1)! * n & \text{si } n > 0. \end{cases}$$

2. *Écrire un programme qui*
   - *Déclare un entier positif x et l'initialise à* 4*,*
   - *Affiche le résultat de* $x!$*.*
3. *Représentez la pile d'appel à l'exécution de votre programme.*

▸ Exercice 2. Addition

1. *Écrire une fonction récursive* `addition` *de deux entiers positifs ou nuls* x *et* y*, en utilisant uniquement les opérations* $+1$ *et* -1*. On rappelle que*

$$x + y = \begin{cases} x & \text{si } y = 0, \\ (x+1) + (y-1) & \text{si } y > 0. \end{cases}$$

2. *Écrire un programme qui calcule et affiche la somme de 4 et 3.*
3. *Représentez la pile d'appel à l'exécution de votre programme.*

▸ Exercice 3. Multiplication

1. *Écrire une fonction récursive* `multiplication` *de deux entiers positifs ou nuls* x *et* y*, en utilisant uniquement la fonction récursive* `addition` *et la décrémentation.*
2. *Écrire un programme qui calcule et affiche le produit de 3 et 2.*
3. *Représentez la pile d'appel à l'exécution de votre programme.*

▸ **Exercice 4. Itératif ou récursif**

1. *Écrire les fonctions itératives* `addition_it` *et* `multiplication_it`*.*
2. *Représentez la pile d'appel à l'exécution des programmes obtenu en remplacant vos fonctions récursives par vos fonctions itératives. Quelle méthode est d'après vous la plus efficace ?*

▸ **Exercice 5. Fibonacci**

1. *Écrire une fonction récursive* `fibonacci` *qui calcule le* n*-ième nombre de fibonacci, noté* f_n*. On rappelle que*
$$f_n = \begin{cases} 1 & \text{si } n = 0 \text{ ou } n = 1, \\ f_{n-1} + f_{n-2} & \text{si } n \geq 2. \end{cases}$$
2. *Écrire la même fonction en itératif.*
3. *En utilisant un exemple et la représentation par la pile d'appel, trouver laquelle des deux implémentations est la plus efficace.*

▸ **Exercice 6. Puissance**

Soient a *un entier et* n *un entier positif ou nul. On se propose de calculer* a^n *en utilisant les trois propriétés suivantes :*

1. $a^n = \begin{cases} a * a^{n-1} & \text{si } n > 0, \\ 1 & \text{si } n = 0. \end{cases}$
2. $a^n = \begin{cases} (a^2)^{\lfloor n/2 \rfloor} & \text{si } n > 0 \text{ est pair}, \\ a * (a^2)^{\lfloor n/2 \rfloor} & \text{si } n \text{ est impair}, \\ 1 & \text{si } n = 0. \end{cases}$
3. $a^n = \begin{cases} (a^{\lfloor n/2 \rfloor})^2 & \text{si } n > 0 \text{ est pair}, \\ a * (a^{\lfloor n/2 \rfloor})^2 & \text{si } n \text{ est impair}, \\ 1 & \text{si } n = 0. \end{cases}$

Écrire, pour chacune des trois propriétés, une fonction récursive qui retourne a^n*.*