Travaux Pratiques d'algorithmique n°3

Cours d'Informatique de Deuxième Année

—DEUG Sciences S3-MI—

Entrées/Sorties et récursivité

La programmation en C d'"Entrées/Sorties"(`printf`, `scanf`) est parfois difficile. Les exercices suivants aident à mieux comprendre certains aspects de la problématique. On verra également quelques fonctions récursives.

▸ **Exercice 1.** *Les coefficients binomiaux sont donnés par la définition récursive suivante :*

$$C_n^p = \begin{cases} 0 & \textit{si } n < p \\ 1 & \textit{si } p = 0 \textit{ ou } p = n \\ C_{n-1}^{p-1} + C_{n-1}^{p} & \textit{si } 0 < p < n. \end{cases}$$

La fonction dont le prototype suit calcule les coefficients binomiaux :

```
unsigned long Cnp(unsigned short n, unsigned short p)
```

Écrire le corps de la fonction `Cnp`*, de deux manières : une récursive et une itérative. Écrire une fonction qui affiche le triangle de Pascal comme dans l'exemple suivant (chaque nombre est affiché sur 5 chiffres) :*

```
    1
    1    1
    1    2    1
    1    3    3    1
    1    4    6    4    1
    1    5   10   10    5    1
    1    6   15   20   15    6    1
    1    7   21   35   35   21    7    1
```

On utilisera le format de `printf` *pour spécifier le nombre de chiffres sur lesquels afficher un nombre. Le transtypage est interdit, ainsi que les conversions de type implicites.*

▸ **Exercice 2.** *Écrire une fonction qui demande à l'utilisateur de rentrer au clavier une expression de la forme "*`+ nombre nombre nombre`*".* ***La saisie se fera en une seule instruction.*** *Cette fonction affichera les 3 nombres séparés par des "*`+`*", suivis du signe "*`=`*" et de la somme des 3 nombres.*
Par exemple : `000003 + -00015 + 000123 = 000111`.
Chacun des nombres devra être affiché sur 6 caractères (y compris un éventuel signe), en comblant éventuellement avec des 0 à gauche. L'exemple précédent correspond à l'entrée suivante : `+ 3 -15 123`.

▸ **Exercice 3.** *Que se passe-t-il si l'utilisateur rentre au clavier l'expression* `+ 3 -t 123` *? Afficher la valeur de retour du* `scanf`*. Expliquer.*

▸ **Exercice 4.** ***(Plus Grand Commun Diviseur)*** *Dans cet exercice, nous nous proposons de calculer le plus grand commun diviseur (**PGCD**) de deux nombres entiers par l'algorithme d'Euclide (on suppose que les entiers ne sont pas tous les deux nuls) :*

$$PGCD(a,b) = \begin{cases} a & \text{si } b = 0, \\ PGCD(b, a \bmod b) & \text{sinon.} \end{cases}$$

1. *calculer le PGCD de* 16 *et* 6 *;*
2. *écrire une fonction récursive qui calcule le **PGCD** de deux entiers (supposés non nuls).*

▸ **Exercice 5. (une multiplication amusante)** *Dans cet exercice, nous cherchons à réaliser la multiplication de deux entiers* `n` *et* `m`*. Malheureusement, on ne dispose que de la division par deux et de l'addition… Cependant, on peut remarquer ceci :*

$$\begin{array}{rcll} 0 * m &=& 0 & \text{pour tout } m, \\ n * m &=& (n/2) * (m+m) & \text{si } n \text{ est pair,} \\ n * m &=& (n-1) * m + m & \text{si } n \text{ est impair.} \end{array} \qquad (1)$$

- *Déduire des relations (1) un algorithme pour réaliser la multiplication de deux entiers ;*
- *en déduire une fonction* `Multiplication` *prenant en arguments deux entiers* `n` *et* `m` *et qui implante l'algorithme précédant ;*
- *évaluer la complexité de la fonction* `Multiplication`*;*
- *écrire une version itérative de cette multiplication.*

Dans les exercices suivants, t est un tableau d'entiers trié pour l'ordre croissant et x est un entier.

▸ **Exercice 6.** *Écrire une fonction itérative qui recherche si x est présent dans le tableau t. Évaluer la complexité de cette fonction en nombre de comparaisons.*

▸ **Exercice 7.** *L'idée de la recherche dichotomique (que l'on ne peut appliquer qu'à un tableau trié) est la suivante : on compare x avec l'élément du milieu du tableau. Si x est présent dans le tableau alors, si x est l'élément du milieu on l'a trouvé et on arrête la recherche. Sinon, il se trouve soit à gauche de l'élément du milieu soit à droite, suivant que x est plus petit ou plus grand que cet élément. Il suffit alors de recommencer la recherche sur une seule de ces deux moitiés de tableau.*

- *Quel est le critère d'arrêt dans la méthode décrite ci-dessus ? Écrire une fonction récursive qui recherche si x est présent dans le tableau en utilisant cette méthode. Quelle est sa complexité ?*
- *Écrire une version itérative de cette fonction. Quelle est sa complexité ?*