TP 1

Juliusz Chroboczek et Gabriel Scherer

14 octobre 2014

Les feuilles de TP sont disponibles sur la page

`http://www.pps.univ-paris-diderot.fr/~jch/enseignement/eidd/`

Avant de commencer ce TP, créez un fichier `~/.emacs` qui contient la ligne suivante :

```
(setq-default c-basic-offset 4)
```

Quittez et relancez Emacs.

Il est *obligatoire* de compiler avec l'option « `-Wall` ». Un exercice n'est pas terminé tant que le programme ne compile pas avec « `-Wall` » sans faire d'avertissements.

Exercice 1 (Un premier programme C)**.** Écrivez un programme C qui affiche « `Bonjour.` ». Compilez-le depuis la ligne de commande, et exécutez le binaire résultant. Compilez-le aussi depuis Emacs (`M-x compile`).

Dans la suite de ce TP, nous vous laissons libres de compiler depuis la ligne de commande ou depuis Emacs, selon vos préférences.

Exercice 2 (Conditionnelle)**.** Écrivez un programme C qui demande à l'utilisateur le premier chiffre de son numéro de sécurité sociale, puis affiche « `Bonjour monsieur.` » si c'est 1, « `Bonjour madame.` » si c'est 2, et « `Bonjour alien.` » sinon. N'utilisez pas l'instruction `switch`. Compilez et testez votre programme.

Exercice 3 (Boucles simples)**.**

1. Écrivez un programme `vertical.c` qui demande à l'utilisateur un entier n et affiche une colonne de « $\star$ » de longueur n. (Je ne mentionne plus, désormais, qu'il faut compiler et tester tous vos programmes.)
2. Écrivez un programme `horizontal.c` qui demande à l'utilisateur un entier n et affiche une ligne de « $\star$ » de longueur n.
3. Écrivez un programme `carres.c` qui affiche les carrés des 10 premiers nombres naturels, c'est-à-dire la suite d'entiers $1, 4, 9 \ldots 100$.

Exercice 4 (Suite de Fibonacci)**.** Leonardo di Pisa, dit Fibonacci, élève des lapins. Les lapins de Fibonacci se comportent de façon un peu particulière : sa première année, un lapin n'est pas adulte, et ne se reproduit donc pas. À partir de ses deux ans, un lapin est adulte et produit chaque année un nouveau lapin. Autrement dit, chaque année, Fibonacci a tous les lapins de l'année précédente, plus autant de nouveaux lapins que de lapins adultes – ceux qu'il avait déjà deux ans avant. Ses lapins ne meurent jamais.

La *suite de Fibonacci* (f_n) est définie par la récurrence suivante :

$$\begin{aligned} f_0 &= 1; \\ f_1 &= 1; \\ f_n &= f_{n-2} + f_{n-1} \qquad \text{pour } n \geq 2. \end{aligned}$$

Écrivez un programme itératif qui s'exécute en espace constant[1] qui lit un entier n puis affiche les n premiers termes de la suite de Fibonacci.

Exercice 5 (Boucles avec accumulation)**.**

1. Écrivez un programme `somme-cubes.c` qui demande à l'utilisateur un entier n puis qui affiche la somme des cubes des n premiers nombres entiers. Par exemple, si l'utilisateur entre 5, votre programme devra afficher 225, car

$$\begin{aligned} \sum_{k=1}^{5} k^3 &= 1^3 + 2^3 + 3^3 + 4^3 + 5^3 \\ &= 1 + 8 + 27 + 64 + 125 \\ &= 225 \end{aligned}$$

2. Écrivez un programme `somme.c` qui demande à l'utilisateur un entier n, puis qui lit n entiers et affiche leur somme et leur moyenne. (Attention — pensez au type des données.)

Exercice 6 (Boucles avec *flags*)**.** Écrivez un programme `sept.c` qui demande à l'utilisateur un entier n puis qui lit n entiers et indique à l'utilisateur si le nombre 7 se trouvait parmi ces n entiers.

Exercice 7 (Boucles, *flags* et fonctions)**.**

1. Écrivez une fonction `premier` qui prend un entier p en paramètre, et retourne 1 si et seulement si p est un nombre premier. Écrivez une fonction `main` qui lit un entier et indique à l'utilisateur si ce nombre était premier.
2. Modifiez votre programme pour qu'il affiche la liste des nombres premiers compris entre 1 et 100[2].

1. Sans utiliser de tableaux.
2. Cet algorithme n'est pas très efficace — nous verrons un algorithme meilleur lors du TP sur les tableaux.