TP N°2
Les boucles

Objectifs :

- ✓ Maitrise des boucles for
- ✓ Maitrise des boucles while
- ✓ Utilisation des tableaux
- ✓ Ecriture et appel de fonctions en langage C

Préambule

Après avoir ouvert virtualBox et vous être logué sous l'environnement CentOS5, on vous demande de créer un répertoire **`tp2`** dans votre répertoire de travail **`algo.`** Vous vous placerez ensuite dans le répertoire **`tp2`** puis créerez 2 fichiers nommés **`tp2_exo1.c`** et **`tp2_exo2.c`**.

1. Les suites (cf TD3)

Moyenne d'une suite

Ecrivez dans le fichier `tp2_exo1.c` un programme qui lit au clavier une suite x0, x1, x2, … des nombres entiers positifs ou nuls et qui en affiche la moyenne. La frappe d'un nombre négatif indique la fin de la série.

On vous demande ensuite de faire une procédure `moyenneSuite()` effectuant le traitement précédent (incluant la saisie des nombres ainsi que l'affichage). Vous sauvegarderez la procédure dans un fichier `suite.c` auquel vous associerez un fichier d'en tête `suite.h`. La procédure `moyenneSuite()` sera appelée depuis la fonction main se trouvant dans le fichier `tp2_exo1.c`

```
> Entrez une suite de nombres (un nombre négatif termine la suite)
> 1
> 4
> 8
> 9
> -1
> La moyenne vaut 5.5
```

Suite inverse

Ecrivez à présent dans le fichier `tp2_exo1.c` un programme qui lit au clavier une suite x0, x1, x2, … de nombres entiers positifs ou nuls et qui les affiche dans l'ordre inverse de leur lecture. La frappe d' un nombre négatif indique la fin de la série. Nous avons des raisons de penser qu'il n'y aura pas plus de 100 nombres. ..

On vous demande ensuite de faire une procédure `inverseSuite()` effectuant le traitement précédent. Vous sauvegarderez la procédure dans le fichier `suite.c` et mettrez à jour le fichier d'en tête `suite.h`. La procédure `inverseSuite()` sera appelée depuis la fonction main se trouvant dans le fichier `tp2_exo1.c`.

```
> Entrez une suite de nombres (max 100 et un nombre négatif termine la
suite)
> 1
> 4
> 8
> 9
> -1
> La série inverse vaut 9 8 4 1
```

2. Les matrices

On supposera des matrices d'Entiers de tailles $M \times N$ avec $M = 4, N = 4$. Une matrice sera représentée en langage C par un tableau d'entiers de taille $M \cdot N$ où l'élément (i, j) de la matrice sera égal à la $(i \cdot N + j)$ème case du tableau comme le montre l'exemple suivant :

$$\begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \Leftrightarrow [a \quad b \quad c \quad d \quad e \quad f \quad g \quad h \quad i \quad j \quad k \quad l \quad m \quad n \quad o \quad p]$$

Affichage de la matrice

On vous demande de créer une fonction `afficheMat()` qui affiche une matrice à l'écran. Vous utilisez la fonction `printf()` associée au symbole `'\t'` permettant de faire une tabulation. Vous sauvegarderez cette fonction dans un fichier nommé `matrice.c` et vous testerez cette fonction dans le fichier `tp2_exo2.c`

```
> La matrice vaut :
> 1     4     2     5
> 3     6     7     0
> 2     8     9     3
> 9     8     1     4
```

Addition de 2 matrices

On vous demande à présent de créer une fonction `addMat()` qui additionne 2 matrices. Vous sauvegarderez cette fonction dans un fichier nommé `matrice.c` et vous testerez cette fonction dans le fichier `tp2_exo2.c`

```
> La matrice A vaut :
> 1     4     2     5
> 3     6     7     0
> 2     8     9     3
> 9     8     1     4
> La matrice A+A vaut :
> 2     8     4     10
> 6     12    14    0
> 4     16    18    9
> 18    16    2     8
```

Multiplication de 2 matrices

On vous demande enfin de créer une fonction `mulMat()` qui multiplie 2 matrices. Vous sauvegarderez cette fonction dans un fichier nommé `matrice.c` et vous testerez cette fonction dans le fichier `tp2_exo2.c`

```
> La matrice A vaut :
> 1    4    2    5
> 3    6    7    0
> 2    8    9    3
> 9    8    1    4
> La matrice AxA vaut :
> 62   84    53    31
> 35   104   111   36
> 71   152   144   49
> 71   124   87    64
```