Méthodes de Conception d'Algorithmes

C. TRABELSI & M. FRANÇOIS

TP8 -- Allocation dynamique de mémoire

EXERCICE (Manipulation de tableaux et matrices)

Partie A (Tableaux)

► **1.** Écrire une fonction `ALLOCATION_TAB_DYN`, qui permet d'allouer dynamiquement de la mémoire à un tableau d'entiers. Elle prend en paramètre la taille du tableau et renvoie un tableau alloué en sortie. Le prototype de la fonction est donné par :

```
int * ALLOCATION_TAB_DYN (int N);
```

► **2.** Écrire une fonction `CHARGEMENT_TAB`, qui permet de charger un tableau d'entiers à partir du flux d'entrée standard. Les caractéristiques du tableau seront décrites dans un fichier ayant la forme suivante :

```
10
7 12 10 94 5 62 145 800 634 4
```

où la première ligne donne la taille du tableau et la seconde les éléments dans l'ordre de stockage. Voilà le prototype de la fonction :

```
void CHARGEMENT_TAB (int N, int * TAB);
```

► **3.** Écrire une fonction `AFFICHAGE_TAB`, qui permet d'afficher les éléments du tableau sur la sortie standard. Sans utiliser les crochets, à chaque étape imprimer l'entier correspondant en utilisant son adresse mémoire. N'oubliez pas de libérer la mémoire pour le tableau à la fin de votre programme.

Partie B (Matrices)

► **1.** Écrire une fonction `ALLOCATION_MAT_DYN`, qui alloue dynamiquement de la mémoire à une matrice (tableau de pointeurs de pointeurs) d'entiers. Cette fonction prend en paramètres les dimensions de la matrice à savoir le nombre de lignes (NB_L) et de colonnes (NB_C) et renvoie la matrice en sortie. Le prototype de la fonction est le suivant :

```
int ** ALLOCATION_MAT_DYN (int NB_L, int NB_C);
```

► **2.** Écrire une fonction `CHARGEMENT_MAT`, qui permet de charger une matrice à partir du flux d'entrée standard. Là aussi, les données seront chargées à partir d'un fichier ayant la forme :

```
3 4
1 5 6 8
5 8 4 2
2 0 1 5
```

où la première ligne indique le nombre de lignes et de colonnes de la matrice et les autres donnent les éléments de la matrice. Le prototype de la fonction est donné par :

```
void CHARGEMENT_MAT (int NB_L, int NB_C, int ** MAT) ;
```

► **3.** Écrire une fonction `LIB_MAT`, dont le but est de désallouer de la mémoire à une matrice. Le prototype est donné par :

```
void LIB_MAT (int NB_L, int ** MAT) ;
```

► **4.** Écrire une fonction `AFFICHAGE_MAT`, permettant d'afficher globalement une matrice donnée en paramètre, sur la sortie standard. Le prototype de cette fonction est donné par :

```
void AFFICHAGE_MAT (int NB_L, int NB_C, int ** MAT) ;
```