TP n°4 de langage C : structures, tableaux 2D

En cas de difficulté, demandez un indice ou une explication à l'enseignant. Chaque exercice doit faire l'objet d'une **validation expérimentale poussée**.

Exercice 1 (acquisition)

Réalisez une fonction **`void incDate(date* d)`** qui incrémente la date pointée par d, où la structure **date** est celle définie ci-contre. Cette fonction devra utiliser une fonction **`int nbJours(int mois)`** qui retourne le nombre de jours dans un mois donné (1 pour janvier, 2 pour février…) d'une année non bissextile, ainsi qu'une fonction **`int bissextile(int annee)`** qui retourne un Booléen indiquant si l'année passée en paramètre est bissextile. Vous devez implanter également ces deux fonctions annexes.

Réalisez une variante **`date incDate( date d)`** qui retourne la date du jour suivant la date passée en paramètre, sans modifier cette dernière.

Donner un exemple d'utilisation de chacune des variantes qui affiche toutes les dates comprises entre deux dates saisies au clavier.

Exercice 3 (acquisition)

Définissez une structure matrice permettant la représentation d'une matrice de valeurs de type double. Cette structure doit contenir le nombre de ligne de la matrice, son nombre de colonnes, et tous ses éléments.

Réalisez une fonction **`matrice creeMatrice( int n, int m)`** qui crée une nouvelle matrice de n lignes et m colonnes avec tous ses éléments initialisés à 0.0.

Réalisez une fonction **`void getVal( matrice* m, int i, int j, int x)`** qui assigne la valeur x à l'élément de la matrice situé en ligne i et colonne j.

Réalisez une fonction **`void printMatrice( matrice* m)`** qui affiche à l'écran la matrice pointée par m.

Réalisez une fonction **`matrice produit( matrice a, matrice b)`** qui crée une nouvelle matrice représentant le produit des deux matrices a et b.

Testez les fonctions réalisées en créant deux matrices, en les remplissant de valeur, en en calculant le produit et en affichant les valeurs des matrices et de leur produit.

```c
typedef struct
{
    int jour;
    int mois;
    int annee;
}date;
```

Exercice 2 (acquisition)

Réalisez une fonction **`int compDates( date* d1, date* d2)`** qui compare les deux dates pointées par d1 et d2 et retourne une valeur négative si la première est antérieure à la seconde, 0 si les deux dates sont identiques, et une valeur positive sinon.

Exercice 4 (consolidation)

Réalisez une fonction **`void tri(date t[], int n)`** qui trie un tableau de n dates en utilisant la méthode du tri fusion vue en TD.

Pour des raisons d'efficacité, le tri complet devra s'exécuter en trois phases :

1. Une phase de création d'un tableau temporaire contenant des pointeurs sur les dates du tableau t.
2. Le tri du tableau temporaire à l'aide d'une fonction de tri appropriée prenant en paramètre un tableau de pointeurs sur des dates.
3. La réorganisation des dates du tableau t.