Connaissances de base sur le langage C (Suite)

Exercice 1 :

Récursivité

1.1. Ecrire la fonction **`fact1`** qui calcule la factorielle de façon itérative.

1.2. Ecrire la fonction **`fact2`** qui calcule la factorielle de façon récursive.

1.3. Ecrire un programme qui calcule le déterminant d'une matrice carrée (N, N), sachant qu'il vaut la somme (sur chaque ligne) de l'élément de la ligne en 1ère colonne par le déterminant de la sous-matrice obtenue en enlevant la ligne et la 1ère colonne (en changeant le signe à chaque fois). Le déterminant d'une matrice (1, 1) est son seul élément. On utilisera bien évidement la récursivité.

1.4. La suite de Fibonacci :

Voici la définition de la suite de Fibonacci:

$$\begin{cases} u_0 = 0,\ u_1 = 1 \\ \forall n > 0,\ u_{n+2} = u_{n+1} + u_n \end{cases}$$

A partir du rang 2, chaque terme est égal à la somme des deux précédents rangs.

I. *Méthode naïve : écrire une fonction récursive* **`int fibo1(int n)`** *basée sur la définition. Tester votre fonction dans un programme principal.*

II. *Récursivité rapide : une nouvelle implémentation va permettre de gagner en temps de calcul. Cette implémentation est basée sur les deux propriétés suivantes de la suite de Fibonacci :*

$$\begin{cases} \forall n > 1,\ u_{2n+1} = (u_n)^2 + (u_{n+1})^2 \\ \forall n > 1,\ u_{2n} = (u_n)^2 + 2u_n u_{n-1} \end{cases}$$

Ecrire la fonction récursive **`int fibo2(int n)`** avec la nouvelle formule. Comparer les temps d'exécution entre **`fibo1`** et **`fibo2`**. Voici comment vous pourrez calculer le temps d'exécution :

```
clock_t start, end;
double elapsed;
start = clock();                                    /* Lancement de la mesure */
/* ... */                                           /* Faire quelque chose    */
end = clock();                                      /* Arrêt de la mesure     */
elapsed = ((double)end - start) / CLOCKS_PER_SEC;   /* Conversion en secondes */
```

La fonction clock est définie dans la bibliothèque time.h.

Exercice 2 :

Structure de type enregistrement

2.1. Ecrire la déclaration d'une structure qu'on appellera **`Un_Tableau_Entier`** contenant un tableau de taille **`NMAX`** entiers et une variable entière nommée **`ncase`** donnant le nombre de cases réellement utilisées dans le tableau; **`ncase`** sera initialisée avec la valeur 0.

2.2. Définir une structure menu comme étant un tableau (d'au plus 20 items) de chaîne de caractères (d'au plus 60 caractères). La structure contiendra aussi un entier **`n`** représentant le nombre d'items du menu.

2.3. Définir le type structuré d'un tonneau, sachant qu'un tonneau étant bombé, il possède un petit diamètre d (les deux bouts), un grand diamètre D (partie ventrue) et une longueur L. Enfin un tonneau contient quelque chose (du vin, du vinaigre, de la bière ou de l'huile). On utilisera la syntaxe avec **`typedef`**.

2.4. Le volume d'un tonneau est donné par la formule suivante :

$$V = 3.14159\ L\ (d/2+2/3(D/2-d/2))2$$

Écrire une fonction qui calcule le volume d'un tonneau en utilisant les deux notations d'accès aux champs d'une structure (**`struct.champ`** et **`*ptrStruct -> champ`**).