TP 7 - les tableaux à 1 dimension

Exercice 1

On considère les déclarations suivantes :

```
#define N 5
typedef int t_unTableau[N] ;
```

1) Écrivez une fonction `int somme(t_unTableau tablo)` qui retourne la somme des N éléments de *t*.

Note : le paramètre formel tableau peut être déclaré comme *t_unTableau* (dans cet exemple) ou bien comme int[] ou bien comme int* ; dans ces deux derniers cas, il faudra aussi passer le nombre d'éléments en paramètre.

2) Écrivez un *main()* testant la fonction précédente sur un tableau *monTableau*, défini et initialisé dans le *main()* comme suit :

```
t_nTableau monTableau = {5,8,2,3,4};
```

On affichera la valeur attendue et la valeur obtenue.

Exercice 2

On considère les déclarations suivantes :

```
#define N 5
typedef int vecteur[N] ;
```

`void remplirVecteur (vecteur v) { …}` // initialise *v* avec les valeurs fournies au clavier .

`void afficherVecteur (vecteur v) { ….}` //affiche à l'écran les N coefficients du vecteur *v*.

`void sommeVecteur (vecteur v1, vecteur v2, vecteur vSomme) {...}` // met dans *vs* la somme des vecteurs *v1* et *v2*.

Complétez les procédures et écrivez le *main()* qui remplit deux vecteurs, en fait la somme et affiche le vecteur résultant à l'écran.

Note : le paramètre formel de type tableau peut être déclaré comme *vecteur* (dans cet exemple) ou bien comme int[] ou bien comme int* (*cf* exercice 1)

Exercice 3

On veut connaître la répartition des notes du DS d'algorithmique et programmation, c'est–à-dire le nombre de 0, le nombre de 1, …, le nombre de 20, ayant été attribués à ce DS.

Question / Écrivez un programme, **utilisant les procédures demandées ci-dessous**, lisant au clavier une suite de notes (une note est un entier compris entre 0 et 20) terminée par –1, et affichant le nombre d'occurrences de chacune des 21 notes possibles (pour améliorer la lisibilité du résultat, on ne fera un affichage que pour les notes ayant au moins une occurrence).

N.B. Les données ne sont pas stockées dans un tableau, mais traitées au fur et à mesure de leur lecture.
On utilisera un tableau *nb* tel que *nb[note]* sera le nombre d'occurrences de la note *note*.

Le programme devra utiliser des procédures :

- pour initialiser le tableau *nb*
- pour mettre à jour le tableau *nb*
- pour afficher les résultats

Exercice 4

Les chaines de caractères sont en fait des tableaux de caractères, à ceci près qu'un caractère spécial (le caractère nul '\0') est toujours présent après le dernier caractère de la chaine. Par exemple, la variable chaine déclarée comme ceci :

```
typedef char tChaine[11];
tChaine ch = "TOTO";
```

contiendra en fait ceci :

0	1	2	3	4	5	6	7	8	9	10
T	**O**	**T**	**O**	**\0**						

Question 1

Écrivez une fonction `int longueur(tChaine ch)` qui calcule et retourne la taille de la chaine de caractères reçue en paramètre (l'équivalent de `strlen()`).

Question 2

Écrivez une procédure `void copie(tChaine ch1, tChaine ch2)` qui copie un à un les caractères de ch2 dans ch1 (l'équivalent de `strcpy()`).

Question 3

Écrivez une fonction `bool compare(tChaine ch1, tChaine ch2)` qui retourne :

- `true` si ch1 et ch2 sont identiques,
- `false` dans le cas contraire.

Exercice 5

Au guichet de la banque X, les clients retirent de l'argent liquide.
La somme en Euros est remise au client en pièces et billets, en minimisant leur nombre.

Question / Écrivez un programme, **utilisant les procédures demandées ci-dessous**, qui pour chaque client demande son nom et la somme que celui-ci retire. La fin de la saisie est indiquée par la présence d'une étoile à la place du nom.

Le programme affiche , pour chaque client, au fur et à mesure de la saisie, le nombre de pièces et billets de chaque sorte à lui remettre, et pour l'ensemble des clients le total de chaque sorte de pièce et billet.

N.B. Les données ne sont pas stockées dans un tableau, mais traitées **au fur et à mesure** de leur lecture.

Les tableaux nécessaires (et suffisants) sont donc :

- une constante tableau *valeurs* contenant les valeurs des pièces et billets.
- un tableau *total* tel que *total[i]* sera le nombre total de pièces ou billets de valeur *valeurs[i]* pour l'ensemble des clients.

Le programme devra utiliser des procédures :

- pour initialiser le tableau *total*
- pour décomposer la somme (avec affichage de la décomposition) et donc mettre à jour le tableau *total*
- pour afficher le récapitulatif pour l'ensemble des clients

On considère les pièces ou billets suivants :
1, 2, 5, 10, 20, 50, 100, 200, 500 .

Exemple d'exécution souhaitée (en gras les valeurs saisies par l'utilisateur) :

```
Nom ? : Toto
Somme ? : 840

Toto :
      1 x 500
      1 x 200
      1 x 100
      2 x 20

Nom ? : Lili
Somme ? : 1750

Lili:
      3 x 500
      1 x 200
      1 x 50

Nom ? : Momo
Somme ? : 18

Momo :
      1 x 10
      1 x 5
      1 x 2
      1 x 1

Nom ? : *

Total pour l'ensemble des clients :
      4 x 500
      2 x 200
      1 x 100
      1 x 50
      2 x 20
      1 x 10
      1 x 5
      1 x 2
      1 x 1
```

Exercice complémentaire

Exercice 6

On considère ici deux tableaux *score1* et *score2* contenant les scores de manches de tennis de table entre deux joueurs (le joueur 1 et le joueur 2). Les deux joueurs ont disputé 4 manches en tout et score1[i] (respectivement score2[i]) est le score du joueur 1 (respectivement joueur 2) pour la manche numéro i.

Question 1/ Proposez une définition pour ces deux tableaux et proposez un programme pour saisir le score des 4 manches.

Question 2/ Complétez le programme précédent en vérifiant si les scores saisis sont corrects (un score est correct si l'un des deux joueurs a atteint au moins 11 points et l'autre moins de 10 points ou que tous les deux ont atteint ou dépassé 10 points avec 2 points d'écart).

Vous procéderez comme suit:

- proposez et testez une fonction booléenne qui admet le score d'une rencontre (donc le score du premier joueur et le score du deuxième joueur) en paramètre, et qui délivre vrai si le score est correct, faux sinon,
- utilisez cette fonction pour détecter les scores incorrects. On affichera chaque score incorrect.