AP1 TP

6 Semaine 6 : Tableaux 1/2

Rappels En C++, les tableaux regroupent sous une même entité un ensemble de variables de même type (entier, caractère ...). La taille d'un tableau doit toujours être définie par une constante :

```
const int TAILLE = 100;
```

Il existe deux façons de déclarer un tableau :

1. déclaration directe :
   `float tab [TAILLE];`
2. déclaration d'un type tableau, puis d'un tableau :
   `typedef float TTabFloat[TAILLE];`
   `TTabFloat tab;`

Les éléments du tableau sont accessibles par l'instruction `tab[i]` pour $i \in [0, TAILLE - 1]$.

En programmation, lorsque l'on passe un **tableau** en paramètre d'une fonction, il est passé **par défaut en paramètre d'entrée/sortie**.

Exercice 1 : Tableau, manipulations de base

1. Recopiez dans votre répertoire tous les fichiers se trouvant dans `/net/Bibliotheque/AP1/TPSem06`. Consultez, compilez et exécutez les fichiers `tableau1.cc` et `tableau2.cc` qui illustrent respectivement les deux déclarations possibles 1. et 2. pour un tableau décrites ci-dessus. Continuez la suite du TP à partir du fichier `tableau1.cc`.
2. Modifiez l'action `saisirTableau` pour qu'elle permette de saisir $taille$ notes d'examen (entiers compris entre 0 et 20) et les place dans un tableau. Attention, vérifiez bien, pour chaque note saisie, qu'elle est comprise entre 0 et 20 (inclus). Si tel n'est pas le cas, demandez à l'utilisateur de recommencer la saisie de la note.

Tests	Résultat(s) attendu(s)	Résultat(s) observé(s)
-1 ;0 ;2 ;0 ;21 ;10 ;20		

   La saisie répétée d'un grand nombre de valeurs (ici 100 au maximum) peut-être pénible, nous vous conseillons pour la suite de vos tests de créer des fichiers contenant les valeurs d'entrée (voir exemple Notes.txt sous `/net/Bibliotheque/AP1/TPSem06/Notes.txt` qui contient 100 notes) et de passer ce fichier en paramètre de votre programme au moment du lancement :
   `>./saisieNotes < Notes.txt`
3. On vous demande de poursuivre le programme précédent en affichant maintenant l'entier le plus grand saisi, ainsi que son indice dans le tableau. Vous écrirez pour cela la fonction suivante qui prend en entrée le tableau et sa taille effective et retourne l'indice de la case qui contient la plus grande valeur :
   `int indiceMax(int tab[ ], int taille)`

Tests	Résultat(s) attendu(s)	Résultat(s) observé(s)
0 ;20 ;0 ;10 ;18		

4. écrivez une action paramétrée réalisant le même traitement que la fonction précédente. Elle aura l'entête suivante :
   `void indiceMax2(int tab[ ], int taille, int & imax)`
   Remarquez le `&` pour le paramètre `imax`.

Tests	Résultat(s) attendu(s)	Résultat(s) observé(s)
0 ;20 ;0 ;10 ;18		
valeur de imax au retour dans le main		

5. Cette fois-ci, on vous demande de créer un tableau (que vous appellerez `compteur`) doté d'une taille fixe `21`. Toute cellule d'indice `i` de ce tableau doit contenir le nombre de fois où le nombre `i` a été saisi (parmi les $taille$ entiers saisis). Poursuivez le programme précédent en créant ce tableau compteur, pour cela vous écrirez une nouvelle action dont voici le prototype :

   ```
   void compteOccurrences(int tab[], int taille, int compteur[], int taille_cpt)
   ```

   Exemple : Si vous saisissez les nombres `12 5 12 8 12 7 12`, alors la cellule `compteur[12]` doit contenir la valeur `4`, les cellules `compteur[5]`, `compteur[7]` et `compteur[8]` doivent toutes contenir la valeur `1`, et toutes les autres cellules du tableau `compteur` doivent contenir la valeur `0`.
   Vous pourrez aussi afficher les valeurs de ce tableau.

Tests	Résultat(s) attendu(s)	Résultat(s) observé(s)
12 ;5 ;12 ;8 ;12 ;7 ;12		

6. La médiane d'un ensemble d'entiers est le plus petit entier de cet ensemble tel que la moitié des entiers de l'ensemble sont inférieurs à cet entier. On se servira du tableau `compteur` pour trouver la médiane des entiers du tableau `tab`. écrivez une fonction calculant la médiane :

   ```
   int mediane(int taille, int compteur[], int taille_cpt)
   ```

   Puis vous afficherez la valeur de la médiane de `tab` dans l'action principale.

Tests	Résultat(s) attendu(s)	Résultat(s) observé(s)
12 ;5 ;12 ;8 ;12 ;7 ;12		

Exercice 2 : Fusion de tableaux croissants

L'objectif est de fusionner deux tableaux dans lesquels les éléments apparaissent dans l'ordre croissant de sorte que le tableau résultant soit lui aussi croissant. Si les deux tableaux en entrée sont de tailles respectives n et m alors le tableau résultant doit être de taille $n + m$.

Exemple : Si nous avons `1 3 5 6 9` et `1 2 3 4 11` alors le résultat de la fusion doit être

```
1 1 2 3 3 4 5 6 9 11
```

Ecrivez une fonction permettant la saisie d'un tableau où les éléments apparaissent dans l'ordre croissant.

```
void saisirCroissant(int tab[ ], int n)
```

Puis vous écrirez une fonction affichant la fusion de deux tableaux *croissants* :

```
void fusionner(int tab1[ ], int taille1, int tab2[ ], int taille2, int tab12[])
```

Pour cela on créera un nouveau tableau tab12[] de taille $taille1 + taille2$ que l'on affichera.