IFT1169 – TRAVAIL PRATIQUE #3 – 21 mars 2008

Librairie standard

Mohamed Lokbani

Équipes : le travail peut-être fait en binôme mais vous ne remettez qu'un travail par équipe.
Remise : une seule remise est à effectuer par voie électronique **le dimanche 6 avril 2007, 23h59 au plus tard, sans possibilités de prorogation**.
Conseils : n'attendez pas le dernier jour avant la remise pour engager votre travail. Vous n'aurez pas le temps nécessaire pour le réaliser.

But : vous allez vous familiariser dans ce travail avec les différents outils de la librairie standard. Deux exercices vous sont proposés. Le premier à travers un exemple simple va vous permettre de voir comment concevoir des conteneurs associatifs. Quant au second, il va vous permettre de manipuler des tableaux multidimensionnels (matrices) à l'aide de conteneurs séquentiels.

Énoncé :

Exercice -1- Soit deux conteneurs associatifs : le premier va associer la clé « idCle » à un « Nom », le second va associer cette même clé à un « Salaire ». Écrire un programme qui permet de concrétiser ce qui suit :
- Initialiser ces deux conteneurs avec les valeurs :

Nom	idCle	Salaire
Nelson	50	1200
Xavier	51	1200
Barbara	52	1500

- Afficher en sortie les noms et les salaires triés par nom : {[Barbara : 52], [Nelson : 50], [Xavier : 51]}
- Afficher en sortie les noms et les salaires triés par salaire : {[Nelson : 50], [Xavier : 51], [Barbara : 52]}

Hypothèses :

- Les noms sont uniques.
- Deux personnes peuvent avoir un salaire identique.

Travail à remettre :

- Votre programme doit porter le nom de « **tp3exo1.cpp** » et le rapport « **tp3exo1rap.pdf** ».

Compilation et respect des spécifications	**1**
Codage, commentaires etc.	**2**
Rapport	**1**
Tests	**1**

Exercice -2-

Le but de cet exercice est de lever la limite fixée lors de la déclaration d'un tableau. Pour cela, nous allons utiliser le conteneur séquentiel « vector ». Par exemple au lieu de déclarer un tableau de 10 entiers ainsi « int tableau[10] », nous pouvons le faire en utilisant le conteneur séquentiel « vector » comme suit « vector<int> v(10) ». La seconde écriture nous permet d'agrandir le tableau sans trop de difficulté en utilisant la méthode « resize » associée à la classe « vector » : « v.resize (v.size( ) + 100) ». Pour libérer l'espace alloué au vecteur « v », il suffit d'utiliser la méthode « clear » associée à la classe « vector » : « v.clear( ) ».

Nous allons généraliser cette approche pour des tableaux à 2 et à 3 dimensions. Un tableau à 2 dimensions est un tableau de tableaux.

[0,0]	[0,1]	[0,2]

Une ligne contient une série de cases dont chacune contient une colonne. Nous avons donc un tableau de tableaux, donc un tableau à 2 dimensions : 1 (ligne) x 3 (colonnes). Comme le premier élément a comme indice [0,0], dans le précédent tableau, l'indice [0,2], correspond à l'élément situé à la 1e ligne et la 3e colonne. Le premier indice représente donc la ligne et le second indice représente la colonne.

[0,0]	[0,1]	[0,2]	[0,3]	[0,4]	
[1,0]	[1,2]	[1,2]	[1,3]		
[2,0]	[2,2]	[2,2]			
[3,0]	[3,2]	[3,2]	[3,3]	[3,4]	[3,5]

Chaque ligne du tableau peut avoir un nombre différent d'éléments.

Un tableau à deux dimensions peut-être représenté par un conteneur séquentiel comme suit :

```
vector < vector < int > > v2d ;
```

Écrire un programme qui permet de concrétiser ce qui suit :

- Déclarer un tableau à deux dimensions.
```
vector < vector < int > > v2d ;
```

- En utilisant la méthode « resize », redimensionner ce tableau afin de contenir **200** lignes par **100** colonnes.
- Initialiser chaque élément de ce tableau avec la valeur « **123** ».
- Afficher tous les éléments du tableau en sortie (l'affichage est libre).
- Libérer l'espace mémoire occupé par ce tableau bidimensionnel.

Nous allons généraliser cette approche à un tableau tridimensionnel. Un tableau tridimensionnel est vu comme un cube ayant une largeur, une hauteur et une profondeur. Nous allons aussi initialiser chaque élément du tableau avec une instance de la classe « vehicule » définie ainsi :

```
class vehicule {
private:
    string nom;
public:
    vehicule() {nom="vw";}
    vehicule(string n_arg):nom(n_arg){}
    void affiche() { cout << "nom du vehicule: " << nom << endl;}
};
```

Écrire un programme qui permet de faire ce qui suit :

- Déclarer un tableau à trois dimensions.
- En utilisant la méthode « resize », redimensionner ce tableau afin d'avoir une largeur de 5, une hauteur de 8 et une profondeur de 10.
- Initialiser chaque élément de ce tableau avec une instance de véhicule précédemment définie.
- Pour chaque élément du tableau, faire appel à la méthode « affiche » de l'instance « vehicule ».
- Libérer l'espace mémoire occupé par ce tableau tridimensionnel.

Travail à remettre :

- Votre programme doit porter le nom de « **tp3exo2.cpp** » et le rapport « **tp3exo2rap.pdf** ».

Compilation, respect des spécifications, codage et commentaires	**3**
Rapport	**3**
Tests	**3**

Remise : il est important de noter que votre TP sera compilé avec « gcc3.4.2 ». Si par choix vous décidez d'utiliser un autre compilateur, vérifiez que le code que vous avez produit (devant normalement fonctionner correctement chez vous) fonctionne aussi bien sur les ordinateurs de la DESI. Pour avoir la version du compilateur, utilisez la commande "**gcc –v**", qui devra donner le numéro de version « **3.4.2** ».
Par ailleurs, assurez-vous de la présence de l'option « pedantic » sur la ligne de compilation. (Cette option n'est pas activée par défaut dans l'utilitaire « devcpp ». Il faudra donc penser à l'activer).

Au total vous devez remettre **4 fichiers**. Pour éviter d'en oublier lors de la remise, nous vous conseillons de regrouper ces 4 fichiers (ou plus) dans un seul fichier compressé **tp3h08.zip** et de remettre ce dernier comme suit :

1. commencez d'abord par vous connecter sur la machine « **remise** » comme il a été pratiqué dans la démo #01.
2. envoyez l'ensemble de vos fichiers par la procédure de remise électronique habituelle (Pour obtenir de l'aide sur cette commande, tapez dans un Xterm : man remise). Respectez les noms des fichiers.

   remise ift1169 tp3 tp3h08.zip

3. vérifiez que la remise s'est effectuée correctement.

   remise –v ift1169 tp3

Barème : ce TP3 est noté sur 14 points : 5 points pour l'exercice -1- et 9 points pour l'exercice -2-.

En plus du précédent barème, vous risquez de perdre des points dans les cas suivants ….

- La non remise électronique (volontaire ou par erreur) est sanctionnée par la note 0.
- Les programmes ne contenant pas d'en-tête, -1 point.
- Un programme qui ne compile pas : 0.
- Un programme qui compile mais ne réalise pas les choses prévues dans la spécification : 0.
- Les avertissements (warnings) non corrigés : cela dépend de la quantité! À partir de -0.25 et plus.
- Le non respect du nom du fichier va générer une erreur de compilation donc un des points de la spécification n'a pas été respecté : 0.
- Aberration dans le codage : même si tous les chemins mènent à Rome, faites l'effort nécessaire pour éviter de prendre le plus long!

Des questions à propos de ce TP?

Une seule adresse : dift1169@iro.umontreal.ca

Pour faciliter le traitement de votre requête, inclure dans le sujet de votre email, au moins la chaîne: [IFT1169] et une référence au **travail03**.

Mise à jour

21-03-2008 diffusion