IFT1169 – TRAVAIL PRATIQUE #3 – 17 mars 2006

« C++ » le restaurateur ! Faire du neuf avec du vieux!

Mohamed Lokbani

Équipes: Le travail peut-être fait en binôme. Vous ne remettez alors qu'un travail par équipe.
Remise : Deux remises à effectuer : électronique et papier le **mardi 04 avril 2006, 22h30 au plus tard, sans possibilités de retard**. Ne seront prises en considération que les copies papiers remises aux démonstrateurs du cours IFT1169 ou bien durant la séance de cours du mardi 04 avril 2006.
Conseils: N'attendez pas le dernier jour avant la remise pour commencer, vous n'aurez pas le temps nécessaire pour le faire.

But : Ce TP est composé de deux exercices. Le premier a pour but de vous faire pratiquer la mise à niveau d'un programme. Ce programme est en rapport avec la surcharge des opérateurs « new » et « delete ». Le second exercice va vous permettre de mettre en pratique des notions apprises sur les classes génériques.

Exercice 1 : Il s'agit de reprendre l'énoncé de la démonstration #7 (Fuite de mémoire), un projet développé par un programmeur il y a de cela quelques années. Les fichiers de ce projet ont été compilés avec une veille version de « gcc ». L'objectif de cet exercice est de mettre à jour ces fichiers pour qu'ils passent la compilation sous la version « 3.2.4 » de « gcc ».

- Vous devez d'abord standardiser le code pour qu'il soit 100% compatible C++ (l'option –pedantic doit être activée).
- Vous devez tenir compte de l'option de compilation « MLEAK » introduite dans le fichier « NEW.H ». Vous devez donc compiler puis exécuter votre programme **avec et sans** cette option.
- Expliquer le coté théorique du programme.
- Expliquer le coté pratique.
- La sortie obtenue suite à l'exécution du programme est donnée dans le fichier « MLEAK.C »

Exercice 2 : Le but de cet exercice est d'écrire **que les méthodes nécessaires** au bon fonctionnement de la classe template « ATrier ».

- La classe template « ATrier » contient la méthode « Tri ».
- La méthode « Tri » accepte deux arguments. Le premier argument est un tableau d'un type quelconque, et le second argument est un entier N, la taille de ce tableau. Cette méthode ne retourne rien. Son rôle est de trier le tableau passé en argument et stocké le résultat du tri dans ce même tableau. Libre à vous d'implanter l'algorithme de tri de votre choix (vous ne devez pas faire appel à un des algorithmes de tri de la STL).

- Quelques exemples d'utilisation de cette classe:

```
int tab[]={1,10,3,5,7};
double xx[]={0.5, 3.2, -6.1, 33.9};

ATrier<int> tInt;
ATrier<double> tDoub;
tInt.tri(tab,5);
tDub.tri(xx,4);
```

- Faites en sorte pour que votre classe puisse traiter aussi la classe « Personne » dont la description est comme suit :

```
class Personne{
public:
      Personne();
      Personne(string Px,string Nx);
      // n==1 pour avoir le prénom,
      // n==2 pour avoir le nom
      string getPersonne(int n);
private:
      string Prenom,Nom;
};

ATrier<Personne> sPersonne;
Personne liste[20];
...
...
sPersonne.tri(liste,20);
```

Les fichiers à remettre :

Version électronique
1) Exercice -1- : Les 3 fichiers corrigés et le rapport.
2) Exercice -2- : « ATrier.h » qui va contenir la classe générique « ATrier ». Le fichier « Personne.h » et « Personne.cpp » pour la définition de la classe « Personne ». Le fichier « exo2.cpp » va contenir la fonction « main » permettant de tester votre programme. Finalement le rapport décrivant votre démarche et l'utilisation de vos programmes.
3) Un seul « Makefile » permettant de compiler et exécuter en fonction de la cible, soit l'exercice 1 seul, soit l'exercice 2 seul, ou bien les deux (i.e. l'un à la suite de l'autre dans une seule opération).

Version papier
Juste la version papier de vos rapports.

Hypothèses et contraintes :

- Pour commencer, vous devez déjà respecter les différentes contraintes de programmation précédemment décrites.
- C'est un travail à faire **SEUL** ! Et seul ne signifie pas deux ni dix. Seul signifie aussi **réfléchir seul**. Plagiat équivaut à un 0 pour commencer.
- IFT1169 est un cours avancé en C++, donc votre programme doit être écrit en C++ et pas en C ni en Java! Nous n'autoriserons aucune référence au langage C. Par exemple l'instruction [#include <stdio.h>] fait référence au langage C, donc non autorisée. Utiliser plutôt [#include <iostream>].
- Vous ne pouvez pas utiliser les « STL » pour concevoir ce travail.
- Le fichier contenant la méthode « **main** », le point d'entrée de votre programme, doit porter le nom « **tp2.cpp** ».
- Les fichiers de sortie ont été générés sous **MinGW/Msys.** Ils sont donc au format DOS étendu. Si vous travaillez sur LINUX, assurez-vous de les convertir au format LINUX en utilisant pour cela la commande **dos2unix**. Pour convertir de LINUX vers le format DOS, utiliser plutôt la commande **unix2dos**. Ces commandes sont disponibles dans un Xterm d'une plateforme LINUX.

Remise : Il est important de noter que votre TP sera compilé avec gcc3.2.4. Si par choix vous décidez d'utiliser un autre compilateur, vérifiez que le code que vous avez produit (qui normalement fonctionne correctement chez vous) fonctionne bien sur les ordinateurs de la DESI. Pour avoir la version du compilateur, utilisez la commande "**gcc –v**", qui devra donnée le numéro de version "**3.2.4**".

Vous devez remettre les fichiers indiqués précédemment et cela avant le mardi 04 avril 2006 à 22h30) :

1. Regrouper d'abord l'ensemble des fichiers à remettre dans un seul fichier « tp3.zip ».

2. Remettre par la suite ce fichier par la procédure de remise électronique habituelle (Pour obtenir de l'aide sur cette commande, tapez dans un Xterm : man remise). Respecter les noms des fichiers.

   remise ift1169 tp3 tp3.zip
3. Vérifier que la remise s'est effectuée correctement.

   remise –v ift1169 tp3

4. Remettez une **copie papier** de vos rapports.

Ne seront prises en considération que les copies papiers remises aux démonstrateurs du cours IFT1169 ou bien durant la séance de cours du mardi 4 avril 2006.

Barème : Ce TP2 est noté sur 12 points.

Exercice -1-	**6**
Exercice -2-	**6**

Détail / Exercice	
Compilation + Codage + Commentaires + Respect des spécifications	**2**
Tests	**2**
Rapport	**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.
- La non remise papier vous pénalise de 2 points.
- Les programmes ne contenant pas d'en-tête, -2 points.
- Un programme qui ne compile pas : 0.
- Un programme qui compile mais ne fait 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 entraîne 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 **tp03**.

Mise à jour

17-03-2006 diffusion