IFT1169 – TRAVAIL PRATIQUE #3 – 14 mars 2009

« La bibliothèque de modèles standard »

Mohamed Lokbani

Équipes : le travail est à faire en monôme (**une seule personne**).
Remise : une seule remise est à effectuer par voie électronique **le mardi 31 mars 2009, 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 : ce TP a pour but de vous faire pratiquer l'utilisation de la bibliothèque de modèles standard (STL) et des fichiers « makefiles ».

Énoncé :

Programme -1- « `tp3exo01H09.cpp` » **:**

Expliquer les différentes instructions du programme et son fonctionnement. Répondre aussi aux différentes questions incluses dans le programme « `tp3exo01H09.cpp` ». Vos réponses doivent être formulées dans le fichier : « `tp3H09.pdf` ».

Programme -2- « `tp3exo02H09.cpp` » **:**

Vous allez jouer un rôle simpliste d'un contrôleur aérien en charge du trafic aérien autour d'une piste donnée. Vous aller gérer une file d'attente d'avions en attente de votre feu vert pour atterrir ou pour décoller de la piste dont vous êtes responsable.
À signaler que nous allons décrire un scénario très loin de la réalité. Le but de ce travail n'est pas de faire de vous des contrôleurs aériens, ni de vous pousser à concevoir des systèmes d'approche pointus. L'objectif est plutôt terre à terre, juste une petite pratique autour la bibliothèque de modèles standard.

Actions : Le programme restera en attente indéfiniment tant que l'une des 3 actions suivantes ne s'est pas produite:

- Ajouter un avion à la file d'attente (a). Cette action correspond à un avion qui demande le feu vert pour atterrir ou pour décoller. Il va donc s'enregistrer auprès de vous. Puis retour en attente.
- Retirer un avion de la file d'attente (r). Cette action correspond à un avion qui a eu le feu vert pour atterrir ou pour décoller. Il va donc être rayé de votre file d'attente. Puis retour en attente.

Attention: Comme cette opération est manuelle, c'est vous qui prenez les décisions! Donc autant prendre la bonne! Un avion en phase d'approche terminale pour l'atterrissage est prioritaire sur un avion en attente du feu vert pour le décollage.

- Quitter l'application (q). Cette action termine l'exécution de votre programme et affiche en sortie tous les avions en attente de traitement (les avions se trouvant encore dans la file d'attente).

Description des Objets

Avion: décrit par son numéro de vol, sa ville de départ s'il s'agit d'un atterrissage ou sa ville de destination s'il s'agit d'un décollage.

Piste: décrit la gestion de la piste. Comme vous devez gérer les décollages et les atterrissages, au moins deux manières de voir les choses (libre à vous de voir les choses autrement):

- Séparation des deux opérations (décollage et atterrissage) donc vous allez prévoir deux conteneurs et un mécanisme en charge de coordonner les ajouts et les retraits. En quelque sorte, vous allez coder le mécanisme de file d'attente.

- Les deux dans le même sac, vous devez démêler les choses pour éviter de faire décoller un avion qui veut atterrir par exemple!

Fichiers fournis : Nous avons fourni le fichier [tp3exo02H09.cpp] qui lance votre application. Vous ne pouvez pas modifier ce fichier en aucun cas.
Pour éviter les problèmes d'inclusion, nous avons inclus le fichier [communs.h] dans [tp3exo02H09.cpp]. Le fichier [communs.h] peut-être modifié à volonté. Il sert à inclure les choses que vous jugez nécessaires pour une compilation correcte de votre programme.
Nous fournissons aussi quelques fichiers d'entrée [in.txt] et de sortie [out.txt] afin de tester votre programme.

Remarques
- C'est un travail sur l'utilisation des « STL », il faudra donc utiliser les différents conteneurs décrits en cours.
- Ce travail se code en quelques lignes donc pas la peine de trop chercher à vous compliquer la vie. Faire d'abord une analyse orientée objet qui est quoi. Bâtir classe par classe et s'assurer que ça marche individuellement avant de connecter les morceaux.
- Comme vous avez affaire à plusieurs fichiers, il est conseillé de procéder en deux étapes pour générer votre exécutable:

-1- compilation individuelle:

```
g++ -pedantic -c toto.cpp -o toto.o
g++ -pedantic -c tata.cpp -o tata.o
g++ -pedantic -c main.cpp -o main.o
```

-2- édition de liens

```
g++ -pedantic -o tp3.exe toto.o tata.o main.o
```

Ou bien (assurez vous de n'a pas avoir des fichiers *.o non nécessaires à votre programme car ici *.o remplace tous les fichiers .o dans votre répertoire)

```
g++ -pedantic -o tp3.exe *.o
```

- En cas de problèmes sur le format des entrées/sorties, utiliser la commande [dos2unix] (pour passer le fichier du format DOS au format UNIX) ou [unix2dos] (l'opération inverse).

Makefile « makefiletp3H09 » **:**

Il vous est demandé de regrouper l'ensemble des commandes de compilation et d'édition de liens dans un seul fichier « makefile ». Pour cela, vous allez définir au moins les trois cibles suivantes :

- « all » : permettant de générer les deux exécutables « tp3exo01H09.exe » et « tp3exo02H09.exe ».
- « ex01 » : permettant de générer l'exécutable « tp3exo01H09.exe ».
- « ex02 » : permettant de générer l'exécutable « tp3exo02H09.exe ».

Remise : il est important de noter que votre TP sera compilé avec « gcc3.4.5 ». 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, veuillez utilisez la commande "**gcc –v**", qui devra donner le numéro de version « **3.4.5** ».

Par ailleurs, assurez-vous de la présence de l'option « pedantic » sur la ligne de compilation.

Fichiers à remettre :

Programme -1-	tp3exo01H09.cpp
Programme -2-	communs.h, vos fichiers

Makefile	`makefiletp3H09`
Rapport	`tp3H09.pdf : décrit le travail réalisé pour les deux exercices et le fichier makefile`

Pour éviter d'omettre un des fichiers associés à ce travail lors de la remise, nous vous conseillons de regrouper l'ensemble des fichiers dans un seul fichier compressé **tp3h09.zip** et de remettre ce dernier comme suit :

1. commencer d'abord par vous connecter sur la machine « **remise** » comme il a été pratiqué dans la démo #01.
2. envoyer 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 tp3h09.zip

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

   remise –v ift1169 tp3

Barème : ce TP3 est noté sur **12 points**. Les points sont répartis comme suit :

Programme -1-	**7.5**
Programme -2-	**3.5**
Makefile	**1**

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 **tp03**.

Mise à jour

14-03-2009 diffusion de l'énoncé.