

TP n°4

Classes

Exercice 1 Donnez un schema UML pour le sujet du TP4 avec les modifications suivantes :

- { les comptes doivent être numerotees de maniere unique ;
- { de même pour les clients. (Un numero de client peut correspondre a un numero de compte, ce n'est pas un probleme) ;
- { un employe peut être charge d'un nombre maximum de comptes, ce maximum est le même pour tous les employes.

Quels sont les constructeurs et methodes qui doivent être publics? Quels sont ceux qui doivent être prives?

Pour quelles classes faudra-t'il prevoir des destructeurs?

Quels sont les champs, methodes, arguments de methodes. . . qui doivent être declares constants?

Reprennez ce que vous aviez fait au TP4 et corrigez vos classes si besoin.

Exercice 2 (Pour ceux qui auraient fini)

Avant de commencer, relisez la partie *La classe d'association* du cours 5.

Dans cet exercice, on modelisera l'association de la classe `Etudiant` a la classe `Cours` par une classe d'association qu'on appelle `Association`. La classe `Cours` possede la methode privee

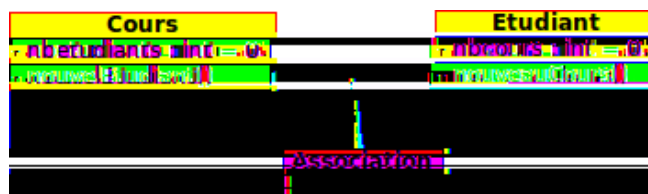


FIGURE 1 { Classe d'association.

`void Cours::nouvelEtudiant()` qui incremente `nbetudiants`. Elle est appelee par la classe `Association` lorsqu'une nouvelle association est creee (similaire pour la classe `Etudiant`).

`Association` a les membres `Etudiant* petudiant` et `Cours* pcours` qui represente cette association. Creez le constructeur `Association::Association(Etudiant *petudiant, Cours *pcours)`. Le constructeur va *notifier* `petudiant` et `pcours` qu'une nouvelle association a ete creee en appelant leurs methodes `nouvelEtudiant()` et `nouveauCours()`. Pensez au concept *friend* lorsque vous appelez ces fonctions.

Pour tester votre programme, creez un tableau d'etudiants et un tableau de cours. Creez quelques associations et verifiez le bon fonctionnement des compteurs `nbcoms` et `nbetudiants`.