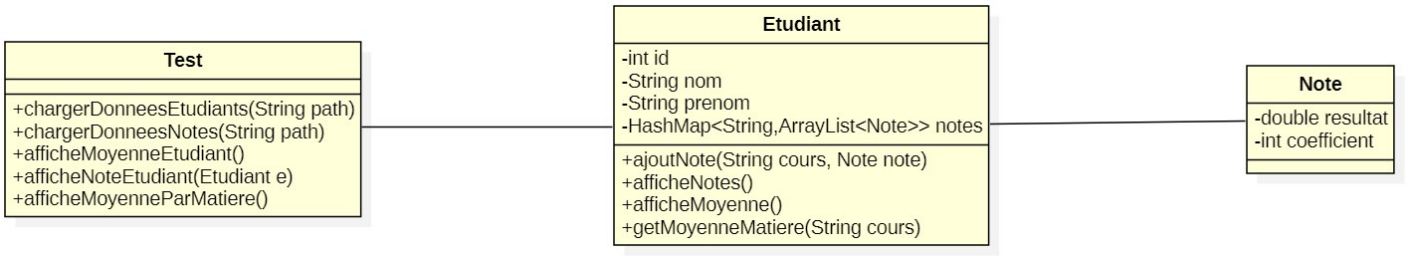
TP07

Notes d'étudiants



Vous avez à votre disposition 2 fichiers csv comportant 2 types de données différentes. Le premier fichier nommé etudiants.csv contient des informations sur les étudiants telles que :

```
idEtudiant;Nom;Prenom
```

```
1;Rousseau;Stephane
2;Jacquot;Aline
3;Timber;Romain
4;Meinier;Estelle
5;Cerizo;Cecilia
```

Le second fichier (notes.csv) comporte des informations sur les notes de ces étudiants sur plusieurs cours. Les données sont représentées ainsi :

```
idEtudiant;Cours;Resultat;Coefficient
```

```
1;Programmation;1;1
2;Programmation;1;1
3;Programmation;1;1
4;Programmation;4;1
5;Programmation;6;1
1;Programmation;3;2
2;Programmation;2;2
3;Programmation;3;2
4;Programmation;3;2
5;Programmation;4;2
1;Management;4;1
```

Vous êtes libre de créer des structures de données (ArrayList, HashMap, HashSet, …) supplémentaires si vous les jugez pertinentes (Spoiler : oui).

La méthode chargerDonneesEtudiants(String path) va se charger de la création des instances d'étudiant à partir du fichier csv. Comme pour le TP précédent, la classe FileToStr est à disposition avec la méthode lireCsv().

La méthode chargerDonneesNotes(String path) va se charger de la création d'instance de Note ainsi que de l'insertion de ces instances dans la bonne instance de Etudiant, dans le bon cours.

La méthode afficheMoyenneEtudiant() va afficher, par étudiant, la moyenne de chaque cours et la moyenne générale de l'étudiant à la fin.

```
Rousseau Stephane
    Modelisation : 3.0
    Management : 4.0
    Reseau : 4.3
    Programmation : 2.3
    Communication : 3.3
Moyenne générale : 3.4
--------------------
Jacquot Aline
    Modelisation : 3.0
    Management : 4.7
    Reseau : 1.7
    Programmation : 1.7
    Communication : 5.3
Moyenne générale : 3.3
--------------------
Timber Romain
    Modelisation : 3.3
    Management : 5.3
    Reseau : 4.3
    Programmation : 2.3
    Communication : 3.7
Moyenne générale : 3.8
--------------------
Meinier Estelle
    Modelisation : 2.8
    Management : 3.0
    Reseau : 4.3
    Programmation : 3.3
    Communication : 4.7
Moyenne générale : 3.6
--------------------
Cerizo Cecilia
    Modelisation : 4.7
    Management : 4.0
    Reseau : 4.7
    Programmation : 4.7
    Communication : 5.3
Moyenne générale : 4.7
--------------------
```

La méthode afficheNoteEtudiant(Etudiant e) va afficher le détail des notes d'un étudiant passé en paramètre. Attention, l'étudiant passé en paramètre ne possède qu'un nom et un prénom (Comme si on avait effectué une recherche), voir méthode main sur Test.

```
Cerizo Cecilia
Modelisation : [5.0 ; 6.0 ; 1.0 ; 5.0]
Management : [4.0 ; 4.0]
Reseau : [4.0 ; 5.0]
Programmation : [6.0 ; 4.0]
Communication : [4.0 ; 6.0]
```

La méthode afficheMoyenneParMatiere() va afficher la moyenne par cours (la moyenne des moyenne d'étudiants).

```
Moyenne par cours :
Modelisation :
3.4
Management :
4.2
Reseau :
3.9
Programmation :
2.9
Communication :
4.5
```

Note 1 : Faites attention au couplage et à la cohésion !!

https://savoirs.usherbrooke.ca/bitstream/handle/11143/1606/MR83692.pdf?sequence=1&isAllowed=y
Je vous épargne la lecture complète :

Cohésion :

2.3 La cohésion appliquée à une classe

Dans la programmation orientée objet, l'unité fondamentale est la classe. Elle combine l'état et le comportement de la représentation d'un concept. La classe est composée des méthodes qui fournissent le comportement et des attributs qui gardent l'état. L'idée est que les méthodes de la classe traitent ses attributs. C'est-à-dire que les méthodes et les attributs devraient avoir un lien étroit. Le degré d'association entre ces composants détermine la cohésion de la classe. Ainsi, [Bieman et Kang, 1995] définissent la cohésion d'une classe comme une caractéristique concernant la connectivité entre les membres de la classe.

Lorsque le degré de cohésion d'une classe est élevé, ses membres sont fortement liés et en théorie la classe devrait décrire de manière appropriée le concept qu'elle représente. Un haut degré de cohésion de la classe est souhaitable, car elle favorise l'encapsulation et la réutilisation. D'ailleurs, la classe avec une cohésion forte est plus facile à comprendre, étant donné qu'elle a un but bien spécifique. À l'inverse, une classe sans cohésion est difficile à maintenir, à comprendre et à réutiliser. Elle est globalement plus fragile et sensible à la variation.

Le manque de cohésion dans une classe peut indiquer la présence d'un problème de conception. C'est un signe que les responsabilités ont été distribuées de manière inappropriée. Il est probable que la classe contient seulement de l'information. Dans tel cas, la classe serait constituée par des attributs et contiendrait peu de méthodes pour traiter les attributs. Au contraire, il est possible que la classe soit constituée par des méthodes qui sont utilisées par d'autres classes.

Couplage :

Les modules qui composent une application logicielle devraient être autant que possible les plus indépendants. La mesure de couplage donne une indication du niveau de cette indépendance. De manière simpliste si toutes les classes d'une application dépendent de toutes les classes constituant l'application, le couplage est trop fort, chaque modification implique de revoir toutes les classes. À l'autre extrême, une application ne peut pas avoir toutes les classes indépendantes, le couplage est nécessaire pour lier quelques classes. Il est considéré que trop de couplage dans une application mène à une complexité accrue du système |Harrison *et al.*, 1998|. Cette complexité peut compromettre la compréhension du module.

Il est plus difficile de détecter des erreurs lorsque les modules sont trop couplés. Il est également difficile de les corriger et de les faire évoluer. En effet, l'interaction d'un module avec plusieurs modules du système le rend plus difficile à maintenir puisque chaque changement subi par le module peut affecter les modules auxquels ce dernier est associé. La complexité d'un système peut être réduite par la conception de modules interagissant faiblement les uns avec les autres.

Note 2 : Les moyennes sont arrondies au dixième. Je vous laisse vous documenter sur comment faire cela. Google est votre pote 😉

Note 3 : Non la moyenne en programmation n'a rien à voir avec votre CC, promis !