TP12 : Java 8 et les Streams (3)

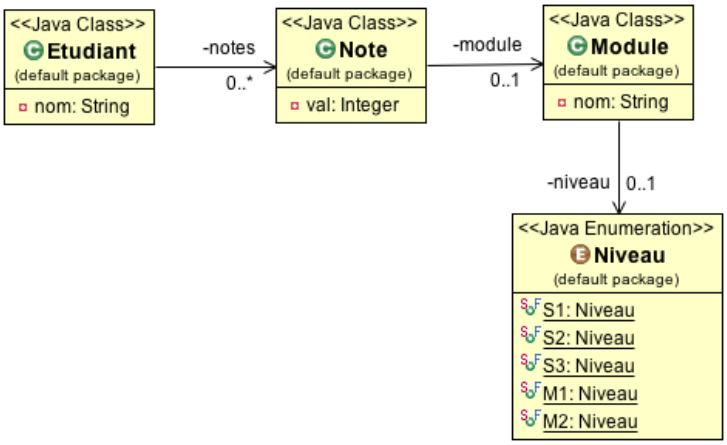
Marianne Simonot

.

Les classes à utiliser

On reprend les classes permettant de modéliser les notes des étudiants se trouvant dans le zip "classesExos2" du dernier TPS sur l'api Collection.

Un étudiant a un nom et une liste de notes.
Chaque note est caractérisée par le module concerné par la note et la valeur de la note. **On suppose qu'il n'y a dans la liste qu'une note par module**. Un module possède un nom et un niveau (S1,S2,S3,M1,M2).



pour simplifier, à la création d'un étudiant, on fournit l'ensemble des notes. On n'aura donc pas besoin d'avoir de méthodes pour ajouter ou modifier une note d'un étudiant. L'initialisation devra ressembler à :

```
Module m11 = new Module("M11", Niveau.S1);
Module m12 = new Module("M12", Niveau.S1);
Module m21 = new Module("M21", Niveau.S2);
Etudiant e1 = new Etudiant("toto", Arrays.asList(new Note(m11, 10),
        new Note(m12, 8), new Note(m21, 12)));
```

exercice 1 : recueillir des informations sur un étudiant

Programmer dans `Etudiant`, en utilisant les streams, les méthodes suivantes :

1. `note(Module m)` qui retourne la note de l'étudiant au module m. retourne -1 s'il n'a pas de note à ce module.
2. `auMoinsUneNoteSous7()` qui teste si l'étudiant a au moins une note au dessous de 7.
3. `aLaMoyennePartout() ()` qui teste si l'étudiant a la moyenne (ie une note supérieure ou égale à 10) à tout ses modules.
4. `aLaMoyenneauxModulesS1()` qui teste si l'étudiant a la moyenne aux modules de S1 (ie une note supérieure ou égale à 10 à chacun des modules de S1).
5. `lesNiveauxIncomplets()` qui retourne les niveaux sans répétition des modules où l'étudiant a une note inférieure à 8.
6. `unNomDeModuleDeS2Rate()` qui retourne l'un quelconque des noms de modules de S2 où l'étudiant a une note sous 10, propage une exception sinon.
7. `moyenne()` qui calcule la moyenne de l'étudiant. Après avoir fait une version utilisant **reduce** et sans chercher à calculer en même temps la somme des notes et le nombre de notes, vous chercherez une autre version en ragardant la javadoc de `Collectors.averingInt`
8. `moyenne(Niveau niv)` qui calcule la moyenne de l'étudiant pour les modules d'un niveau donné.
9. `lesModulesObtenus()` qui retourne les modules où l'étudiant à la moyenne.
10. `meilleureNote()` qui retourne la meilleure note de l'étudiant. Vous donnerez une version sans `Collectors.maxBy` puis une avec. **cf javadoc de Collectors.maxBy**
11. `lesNotesParNiveau()` qui retourne les notes de l'étudiants groupées par niveau sous la forme d'une Map dont les clés sont les niveaux et les valeurs les listes de notes de ce niveau de l'étudiant. **Pour cela vous regarderez la javadoc de Collectors.groupingBy**

exercice 2 : la promo

On s'intéresse maintenant à la promo. Une promo est caractérisée par une collection d'étudiant.

1. Définir dans `Promo` la méthode `lesNomsDesEtudiantsRecus(Module m)` qui retourne les noms des étudiants qui ont une note Supérieure ou égale à 10 au module m.
2. Définir dans `Promo` la méthode `NomMajorS1()` qui retourne le nom de l'étudiant qui a la meilleure moyenne au S1.