Université de Paris **Langages à objets avancé**
M1 **Année 2020—2021**

TP Noté n° 1

Ce travail est à réaliser chez vous et individuellement et à déposer sur Moodle au plus tard le dimanche 4 octobre 23h59 sous forme d'une archive [1].

Un fichier `main.cpp` vous est fournit. Votre code devra pouvoir être exécuté avec la fonction `main` définie dedans.

Réfléchissez bien à ce qui doit être déclaré constant dans votre programme, à ce qui doit être passé par référence en argument de vos méthodes et fonctions. Vous pourrez bien sûr ajouter dans chaque classe les accesseurs et modificateurs si nécessaire.

Gestion de notes

Vous allez écrire un ensemble de classes qui permettent de gérer les notes des étudiants dans différentes u.e. (unité d'enseignement). Un étudiant sera défini par un inscrit et un ensemble de résultats. Un inscrit est défini par son nom, son prénom et son numéro d'étudiant. Un résultat correpond à l'ensemble des notes obtenues dans une u.e. avec le coefficient associé à chaque note pour un étudiant donné. Alors une u.e. est identifiée par un code et regroupe l'ensemble des résultats correspondant à sa matière.

Diagramme UML

Faire le diagramme UML de cette gestion des notes. Attention, nous vous demandons de joindre à votre archive une image de ce diagramme ou un pdf (pas le fichier bouml ou StarUML).

Pour chaque classe définie ci-dessous vous devrez pouvoir afficher une instance en surchageant l'opérateur `<<`.

Inscrit

1. Écrivez une classe `Inscrit` avec trois attributs constants représentant un nom, un prénom et numéro (entier positif). La classe a deux constructeurs, un avec trois arguments représentant le nom, prénom et numéro et l'autre sans argument qui initialise à un inscrit par défaut. Le numéro sera un argument optionnel, initialisé à 0 par défaut.

Note : vous pouvez utiliser la classe `string` pour repésenter une chaîne de caractères.

2. Ajoutez un constructeur de copie.

1. `tar cvf mon_fichier.tar fic1 fic2 ...` pour créer l'archive `mon_fichier.tar` composée des fichiers `fic1`, `fic2`, ...

Resultat

1. Écrivez une classe `Resultat` avec deux attributs constants représentant un code (suite de caractères) et un inscrit et deux attributs représentant un tableau de notes `notes` et un tableau de coefficients `coefs` à valeurs réelles. La classe a un constructeur à deux arguments représentant le code et l'inscrit.
Note : vous pouvez utiliser la classe `vector` pour repésenter un tableau d'éléments.

2. Ajoutez à votre classe `Resultat`, une méthode `moyenne` retournant un réel valant la moyenne des notes pondérées par les coefficients. Par exemple si `notes` vaut $\{17, 8.5, 12\}$ et `coefs` vaut $\{0.2, 0.3, 0.5\}$ alors la moyenne est égale à $17 \times 0.2 + 8.5 \times 0.3 + 12 \times 0.5 = 11.95$ et `moyenne` retourne 11.95.

3. Ajoutez à votre classe, une méthode `ajout_note` qui prend en argument une note `n` et un coefficient optionnel `c` valant 0 par défaut et ajoute à `notes` la note `n` et à `coefs` le coefficient `c`.

UE

1. Écrivez une classe `UE` avec un attribut constant représentant le code de l'UE (suite de caractères) et un attribut représentant un tableau de résultats. La classe a un constructeur à un argument représentant le code.

2. Ajoutez à votre classe `UE`, une méthode `set_coefs` qui définit le tableau de coefficients de chaque entité `Resultat` à partir d'un vecteur de réels passé en argument de la méthode.

3. Ajoutez à votre classe, une méthode `moyenne` retournant un réel valant la moyenne des moyennes de chaque résultat.

4. Ajoutez à votre classe, une méthode `ajout_note` qui prend en argument une note `n`, un numéro d'étudiant `num` et un coefficient optionnel `c` valant 0 par défaut et ajoute à `notes` la note `n` et à `coefs` le coefficient `c` pour l'étudiant de numéro `num`.

5. Ajoutez à votre classe, une méthode `ajout_etudiant` qui prend en argument un inscrit, et ajoute au tableau de résultats un nouveau résultat pour cet inscrit. Attention, à ne pas créer un nouvel inscrit.

Etudiant

1. Écrivez une classe `Etudiant` avec un attribut constant représentant un inscrit et un attribut représentant un ensemble de résultats. La classe a deux constructeurs, un avec trois arguments qui représentent le nom, prénom et numéro de l'étudiant et un avec un argument représentant un inscrit.

2. Ajoutez à votre classe, une méthode `moyenne` qui prend en argument un code d'u.e. et retourne la moyenne de l'étudiant dans l'u.e.

3. Ajoutez à votre classe, une méthode `synchro` qui prend en argument une u.e. et met à jour les résultats de l'étudiant en prenant en compte son résultat dans l'u.e. Attention de bien mettre à jour son tableau de notes et de coefficients.