Associations simples et collections

1 Promotions

Vous avez déjà modélisé et implémenté une classe Etudiant. Nous allons étudier une classe Promotion, qui utilisera la classe Etudiant.

Question 1. Une promotion est un ensemble d'étudiants, pour une année donnée. Nous représentons cette relation entre la classe `Etudiant` et la classe `Promotion` par une association (voir Figure **??**).

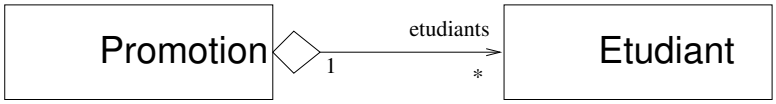

FIGURE 1 – Classes Etudiant et Promotion

a- Mettez en place la classe `Promotion` et l'association entre `Promotion` et `Etudiant`. Pour cela, on placera dans la classe `Promotion` une collection d'étudiants, on choisira la collection ArrayList. Cette collection aura une visibilité privée. On veillera à bien mettre dans la classe Promotion un attribut représentant l'année, les accesseurs associés, et deux constructeurs : un constructeur sans paramètres, et un constructeur prenant une année en paramètre. Les constructeurs initialiseront l'année et créeront la collection. La documentation de la classe ArrayList de Java est disponible via la documentation de l'API Java.

b- Écrire deux méthodes publiques manipulant la collection : une retournant le ième étudiant de la collection, et une retournant le nombre d'étudiants de la promotion.

c- Mettez en place une classe TestPromotion (avec un `main`) qui vous permet de tester votre classe `Promotion`.

Question 2. Complétez la classe `Promotion` avec les méthodes suivantes (on veillera à bien tester chacune des méthodes au fur et à mesure, grâce à la classe `TestPromotion`) :

- une méthode `inscrire` qui permet d'inscrire un étudiant dans la promotion ;
- une méthode `moyenneGenerale` qui retourne la moyenne générale de la promotion ;
- une méthode `afficheResultats` qui affiche une ligne pour chaque étudiant (correspondant au résultat de la méthode `ligneResultat`) ;
- une méthode `recherche` qui permet de retrouver un étudiant d'après son nom. On suppose qu'il n'y a pas d'homonymes ;
- une méthode `admis` qui retourne l'ensemble des étudiants admis ;
- une méthode `populationeIssuDeRecrutementExterieur` qui retourne l'ensemble des inscrits issus d'une recrutement par eCandidat ou études en France ;
- une méthode `majors` qui retourne les étudiants dont la moyenne est la plus élevée (il peut y avoir plusieurs étudiants majors ex-aequo).

2 Utilisation de la classe Promotion

Écrivez une classe avec un `main` permettant d'effectuer les opérations suivantes (cette classe doit fortement ressembler à votre classe de test) :

- création d'une promotion vide d'étudiants,
- inscription des étudiants dans cette promotion (les étudiants doivent bien sûr être créés auparavant),

— affichage du nombre des nouveaux inscrits issu de recrutement extérieur,
— attribution des notes aux étudiants,
— affichage du nom des majors de la promotion,
— affichage des résultats.

3 Transport du courrier

Revenons sur les objets postaux introduits lors des TD-TP précédents. Nous allons maintenant nous intéresser à la description des *sacs postaux*. Un *sac postal* peut contenir un certain nombre d'objets postaux, et dispose d'une capacité maximale. Cette capacité est de $0,5m^3$ pour un sac ordinaire. On peut fabriquer des sacs d'une autre capacité, sur demande précisant la capacité voulue.
On doit pouvoir ajouter un objet dans un sac, s'il y rentre. On veut connaître le volume occupé par un sac (compter forfaitairement $5dm^3$ pour la toile du sac), et sa valeur de remboursement en cas de perte. Enfin, on désire aussi pouvoir remplir un nouveau sac avec tous les objets de même code postal extraits d'un autre.

Question 3. Complétez le diagramme de classes déjà réalisé pour y intégrer la notion de sac postal.

Question 4. Ecrivez et testez la classe Java correspondante.

4 Pour ceux qui auraient fini en avance : Histogrammes

Complétez la classe Promotion de manière à pouvoir :

— afficher un histogramme des moyennes de la promotion. Par exemple, si les étudiants ont obtenu les moyennes suivantes : `Jacques 16,3, Justine 18, Germain 15,7, Hugues 12,2, Sylvia 15,9, Gaston 11,4, Astrid 11, Kim 11,1` , on affiche le diagramme suivant (limité entre 10 et 20 ici) :

```
[10 − 11[
[11 − 12[ ***
[12 − 13[ *
[13 − 14[
[14 − 15[
[15 − 16[ **
[16 − 17[ *
[17 − 18[
[18 − 19[ *
[19 − 20]
```

— connaître les moyennes les plus fréquentes,
— en utilisant l'histogramme, déterminer combien de personnes ont une certaine moyenne (arrondie) donnée,
— afficher le même histogramme qu'à la première question mais pivoté de 90° vers la droite.

```
[10 − 11[  [11 − 12[  [12 − 13[  [13 − 14[  [14 − 15[  [15 − 16[  [16 − 17[  [17 − 18[  [18 − 19[  [19 − 20]
           ***        *           *           **          *           *
```

On veillera à ce qu'il n'y ait pas de duplication de code entre les deux méthodes d'affichage d'histogramme.