Module : POO

TD3 : Eléments statiques d'une classe

Le but de ce TD est d'apprendre à manipuler les attributs et méthodes statiques d'une classe.

Exercice 1

1) Écrivez une classe **Tab** qui contient les attributs et méthodes suivantes :

- ***NMAX*** : une constante statique de type entier égale à 40.
- ***remplir (float t[])*** : méthode statique qui remplit un tableau t par des valeurs saisies au clavier.
- ***somme_element (float t[])*** : méthode statique qui retourne la somme des cases d'un tableau t.
- ***additionner (float t1[]), float t2[])*** : méthode statique qui additionne chaque case du tableau t1 avec une case à la même position du tableau t2 et retourne le tableau résultat.
- ***produit (float x, float t[])*** : méthode statique qui multiplie chaque case du tableau t par x et retourne le tableau résultat..
- ***lister (float t[])*** : méthode statique qui affiche à l'écran les cases d'un tableau t.

2) Un enseignant aimerait calculer automatiquement les moyennes de ses étudiants dans une matière et déterminer la moyenne de la classe, sachant que :

$$\text{moyenne} = (\text{note de contrôle} + 2 * \text{note d'examen})/3$$

Utilisez les méthodes de la classe *Tab* pour écrire une classe ***CalculMoyenne*** contenant la méthode main, dans laquelle :

- Déclarez deux tableaux **noteCtrl** et **noteExam** de **nb** réels, tel que *nb* doit être unentier positif et inférieur à *NMAX*. Puis, remplissez-les au clavier et affichez leurs contenus.
- Calculez la moyenne des nb étudiants (à mettre dans un 3ème tableau *moy*).
- Afficher les moyennes de tous les étudiants ainsi que la moyenne de la classe.

3) L'enseignant voudrait ajouter un point et demi de bonus à tous ses étudiants. Ajoutez à la classe Tab la méthode *additionner(float x, float[])* qui retourne un tableau contenant les cases du tableau t incrémentées de x. Ensuite, appelez cette méthode dans le main de la classe *CalculMoyenne* pour aider l'ensiegnant.

4) Que se passe-t-il si les tableaux noteCtrl et noteExam n'ont pas la même taille? Comment y remédier ?

Exercice 2

1) Ecrivez une classe Java nommée **MotDico** ayant les attributs suivants :

- **num** : numéro du mot, doit être généré automatiquement
- **mot** : de type String
- **definition** : de type String qui représente la définition du mot.

Ajoutez les méthodes suivantes :

- **String getMot ()** : pour retourner le mot.
- **String getDéfinition ()** : pour retourner la définition.
- **void setDéfintion (String s)** : Pour donner ou changer la définition d'un mot.
- **void setMot (String s)** : pour donner ou changer le mot.
- **boolean synonyme (MotDico m)** : retourne vrai si un MotDico est synonyme decelui donné en paramètre.

2) Écrivez une classe nommée **Dictionnaire** décrite par:

- **nbMots** : contient le nombre des mots d'un dictionnaire.
- **dico** : c'est un tableau de MotDico.
- **nom** : c'est le nom de dictionnaire.
- un **constructeur** qui permet de créer le tableau dico et d'initialiser le nom d'un dictionnaire.
- **void ajouterMot (MotDico m)** : Ajoute un mot et sa définition au dictionnaire.
- **void supprimerMot (String ch)** : c'est une méthode qui supprime un mot et sa définition du dictionnaire.
- **int chercherMot (String ch)** : C'est une méthode qui permet de retourner la position d'un mot dans le dictionnaire. S'il n'existe pas, elle retourne -1.
- **void listerDico ()** : permet de lister tout le contenu de dictionnaire.
- **int nbSynonymes (MotDico m)** : retourne le nombre de synonymes d'unMotDico.

3) Écrivez une méthode main dans laquelle :

- Créez un dictionnaire intitulé Larousse
- Ajoutez trois mots au dictionnaire puis lister son contenu.
- Cherchez un mot qui n'existe pas, puis un qui existe et affichez-le avec sa définition.
- Affichez le nombre de synonymes d'un mot puis supprimez-le.