Etablissement : ISET-Charguia	**Département :** Technologies de l'Informatique
Matière : Atelier POO Avancée	**Année :** 2ème année
Année Universitaire : 2023- 2024	

TP n° 3 : Les collections sous JAVA

Objectif du TP :

Utiliser les types de collection : ArrayList et LinkedList

Objectif : Il s'agit d'implémenter une classe **Etudiant** qui désigne un étudiant et une classe **ClasseEtudiants** qui permet de gérer une classe et la collection des étudiants de la classe.

Partie 1 : Utilisation de la collection ArrayList

1) Implémenter la classe **Etudiant** qui renferme :

Les attributs :

- ***nCE*** : numéro de la carte d'étudiant de type String
- ***nom*** : nom de l'étudiant de type String
- ***moyenne*** : Moyenne de l'étudiant de type float.
- ***classeEtudiant*** : La classe de l'étudiant de type ClasseEtudiants

Les méthodes :

- *Constructeur paramétré* acceptant pour paramètre : le numéro de la carte d'étudiant
- *Constructeur paramétré* acceptant pour paramètres : le numéro de la carte d'étudiant, le nom de l'étudiant, et la moyenne de l'étudiant
- Les *accesseurs* et *mutateurs* des différents attributs.
- Redéfinition de la méthode *equals* sachant que deux étudiants sont considérés comme égaux s'ils ont le même nCE.
- Une redéfinition de la méthode *toString* qui renvoie une chaîne de caractères telle sous le format :

« Etudiant : N° carte Etudiant : 1245, Nom : Mohamed Saleh, Classe : DSI2, Moyenne : 12 »

2) Implémenter la classe **ClasseEtudiants** qui renferme :

Les attributs :

- ***nomClasse*** : désigne la désignation de la classe.
- ***listeEtudiants*** de type ArrayList<Etudiant>

Les méthodes

- *Constructeur paramétré* acceptant comme paramètre la désignation de la classe.
- *L'accesseur* à l'attribut nomClasse

- ***ajouterEtudiant*** :

Paramètre	un objet de type Etudiant
Retour	un booléen qui vaut true si l'ajout a réussi false sinon.
Spécification	Permet d'affecter à l'attribut **classeEtudiant** de **Etudiant** la désignation de la classe et d'ajouter l'étudiant à la collection **listeEtudiants**.

- ***ajouterEtudiant*** : //Surcharge

Paramètres	l'index de type int et un objet de type Etudiant
Retour	un entier qui vaut 1 si l'ajout a réussi, 0 si l'étudiant existe déjà dans la liste et -1 si l'index fourni n'est pas dans l'intervalle autorisé.
Spécification	Permet d'affecter à l'attribut **classeEtudiant** de **Etudiant** la désignation de la classe et d'ajouter l'étudiant à la collection **listeEtudiants** à la position passée en paramètre

- ***ajouterAuDebut*** :

Paramètre	un objet de type Etudiant
Retour	un booléen qui vaut true si l'ajout a réussi false sinon.
Spécification	Permet d'affecter à l'attrribut classeEtudiant de **Etudiant** la désignation de la classe et d'ajouter l'étudiant à la collection **listeEtudiants** au début de la collection

- ***chercherEtudiant*** :

Paramètre	le numéro de la carte d'étudiant
Retour	un objet de type Etudiant.
Spécification	Permet de renvoyer l'étudiant de la collection listEtudiants, dont le NCE correspond à celui passé en paramètre. Si aucun étudiant n'est trouvé **null** est renvoyé.

- ***chercherEtudiant*** : //Surchargé

Paramètre	index de type entier
Retour	un objet de type Etudiant
Spécification	Permet de renvoyer l'étudiant de la collection listEtudiants, dont l'index dans la liste est celui passé en paramètre. Si index est hors intervalle **null** est renvoyé.

- ***SupprimerEtudiant*** *:*

Paramètre	le numéro de la carte d'étudiant
Retour	un booléen qui vaut true si l'étudiant est supprimé, et false si l'étudiant n'existe pas dans
Spécification	Permet de supprimer l'étudiant de la collection listEtudiants, dont la carte d'étudiant est fournie en paramètre. La valeur false est renvoyée si l'étudiant ne se trouve pas dans la collection

- ***afficherListeEtudiants*** *:*

Spécification	Permet d'afficher la classe et la liste des étudiants de la classe (un étudiants dans chaque ligne).

- ***viderEtudiants*** :

Spécification	Permet de supprimer tous les étudiants de la liste **listeEtudiants**

3) Ecrire une classe **Program** qui renferme :

- Une méthode statique ***creerEtudiant*** qui permet de saisir les informations de l'étudiant (nce, nom et moyenne) et de renvoyer un objet de type étudiant qui renferme ces informations
- La méthode statique **main** qui permet de :
  - Instancier une classe en lui faisant passer la valeur « DSI2 »
  - Offrir et prendre en charge les fonctionnalités du menu suivant :

---------- MENU----------
1- Ajouter un étudiant
2- Insérer un étudiant à une position donnée
3- Ajouter un étudiant en début de la liste
4- Consulter un étudiant en fournissant son NCE
5- Consulter un étudiant se trouvant à une position donnée
6- Supprimer un étudiant connaissant son NCE
7- Afficher la liste des étudiants
8- Vider la liste
9- Quitter

Partie 2 : Utilisation de la collection LinkedList

Reprendre la partie 1 en utilisant cette fois la collection LinkedList et apporter les modifications nécessaires aux classes.