COLLÈGE MONTMORENCY

Atelier #3 : Les tableaux d'objets

Dans cet exercice, vous allez développer un programme de gestion de produits d'un entrepôt. Votre programme doit stocker les produits dans une liste de produits constituée d'un tableau d'objets de type Produit. Il doit permettre d'effectuer les traitements suivants:

- Initialiser la liste de produits: enregistrer des produits existants dans la liste.
- Afficher les produits de la liste.
- Ajouter un nouveau produit dans la liste. Les informations du produit doivent être saisies à partir du clavier.
- Supprimer un produit de la liste.
- Rechercher un produit dans la liste.

Téléchargez le package **`at03`** se trouvant dans Moodle. Ce package contient 3 classes: `Produit`, `ListeProduits` et `GestionEntrepot`. Copiez ce dossier dans le package portant votre nom dans votre projet **420ZD5**, puis corrigez l'importation de package. Complétez les classes aux endroits indiqués par «`TO DO`» en suivant les directives de cet atelier.

Question 1: La classe `Produit`

Complétez la classe `Produit`. Cette classe représente un produit d'un entrepôt. Vous devez compléter 2 méthodes de cette classe seulement. Mais avant cela, je vous conseille d'analyser le code fourni.

Voici le contenu de cette classe.

Attributs et méthodes existants :

- Plusieurs constantes statiques (voir la classe)
- Un attribut `numero` de type `int`
- Un attribut `description` de type `String`
- Un attribut `prix` de type `double`
- Les méthodes d'accès (getter) de chaque attribut.
- Les méthodes de modification (setter) de chaque attribut.
    - Le numéro de produit : entre 0 et 100000.
    - La description : au minimum 2 caractères et au maximum 250 caractères. Aucun affichage dans cette méthode
    - Le prix : entre 0 et 20000. Aucun affichage dans cette méthode
- Un constructeur avec 3 paramètres.
- Un constructeur par défaut (sans paramètres).

- Une méthode `String toString()` qui retourne une représentation du produit sous la forme : numero description prix.
  - Exemple: `250 Chaise rouge 98.25$`
- Une méthode `public void lireNumero(Scanner clavier)`: elle demande de lire un numéro de produit valide à partir du clavier, puis elle affecte cette valeur à l'attribut `numero`.

Travail à faire:

- Complétez la méthode `public void lirePrix(Scanner clavier)`: elle demande de lire un prix valide à partir du clavier, puis elle affecte cette valeur à l'attribut `prix`.
  Vous devez compléter cette méthode en vous inspirant de la méthode `lireNumero()` ci-haut.
- Complétez la méthode `public void lireDescription(Scanner clavier)`: elle demande de lire une description valide à partir du clavier, puis elle affecte cette valeur à l'attribut `description`.
  Vous devez compléter cette méthode en vous inspirant de la méthode `lireNumero()` ci-haut.
- Complétez la méthode `main()` pour tester les méthodes ci-haut. Ajoutez les instructions pour effectuer les traitements suivants.
  - Créez puis affichez un premier produit avec les informations ci-dessous :
    1 Stylo 12,25$
  - Créez puis affichez un deuxième produit avec les informations provenant du clavier comme indiqué dans la trace d'exécution ci-dessous :

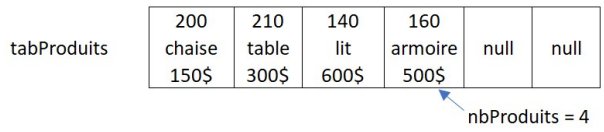
```
Entrez le numéro du produit: 250
Entrez la description du produit: Chaise rouge
Entrez le prix du produit: 98,25
250 Chaise rouge 98,25$ //appelez toSting() pour afficher les
information du deuxième produit
```

Question 2: **la classe** `ListeProduits`

Complétez la classe `ListeProduits`. Cette classe représente une liste de produits représentée par un tableau à une dimension de type `Produit`. Ses méthodes permettent d'effectuer des opérations telles que l'ajout, la suppression et la recherche d'un produit dans le tableau. Cette classe contient les attributs suivants:

- `tabProdui`: un tableau de type Produit.

- `nbProduits` : le nombre de produits qu'il y a dans le tableau (pas le nombre de cases du tableau).



Travail à faire :

Implémentez les méthodes suivantes:

1. Un constructeur `public ListeProduits (int maxProduits)` qui consiste à :
   - initialiser le tableau `tabProduits` avec un nombre de cases égal à `maxProduits`.
   - Initialiser `nbProduits` à zéro
2. Une fonction `public int taille()`: elle retourne le nombre d'éléments se trouvant dans le tableau de produits.

3. Une fonction `public boolean estVide`: elle retourne `true` si le tableau de produits est vide, `false` sinon.

4. Une fonction `public boolean estPlein()`: elle retourne `true` si le tableau de produits est plein, elle retourne `false` sinon.

5. Une méthode `public Produit obtenirProduit(int indice)` qui retourne le produit correspondant à l'indice donné en paramètre. Cette méthode vous est fournie. Vous devez simplement la documenter.

6. Une fonction `public boolean ajouter(Produit nouveauProduit)`: elle ajoute le produit reçu en paramètre dans le tableau de produits s'il n'est pas plein. Elle retourne `true` si le produit est ajouté sinon, elle retourne `false`.
   Voici comment procéder:
   - Utilisez une variable booléenne qui vaut `false` au début. Cette variable prendra la valeur `true` si le produit est ajouté.
   - Appeler la méthode `estPlein()` pour savoir si le tableau n'est pas plein. Si tel est le cas, ajoutez le produit dans le tableau:
     - l'ajout se fait dans la case d'indice `nbProduits`.
     - incrémenter `nbProduits`.
     - affecter `true` à la variable booléenne.

7. Une méthode public `void listerProduits()` : elle affiche les produits contenus dans le tableau. L'affichage devrait être sous forme:

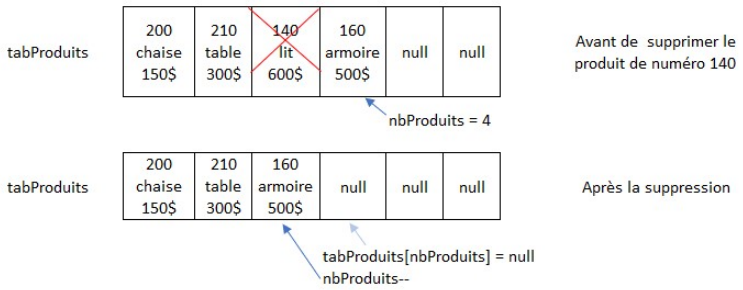
```
100 Chaise          200,00$
200 table           500,00$
300 armoire         1000,00$
```

8. Une méthode `public int trouverProduit(int numeroAChercher)`: elle retourne l'indice du produit, dont le numéro est donné en paramètre, dans le tableau des produits. Elle retourne -1 si le produit n'existe pas.

9. Une méthode `public boolean supprimer(int numeroASupprimer)` : elle supprime le produit dont le numéro est donné en paramètre du tableau. Elle retourne `true` si le produit a été supprimé, `false` sinon.
   Voici comment procéder:
   - Vérifiez d'abord que le tableau n'est pas vide en appelant la méthode `estVide()`.
   - Trouvez l'indice du produit à supprimer dans le tableau en appelant la méthode `trouverProduit()`, soit `indice`.
   - Si indice est différent de -1 alors :
     - décalez le contenu des cases du tableau vers la gauche à partir de `indice+1`.

```
for (int i = indice + 1; i < nbProduits; i++) {
      tabProduits[i - 1] = tabProduits[i];
}
```

     - Mettre `null` dans la case d'indice `nbProduits` (voir la figure ici-bas)
     - Décrémenter `nbProduits`



- Compléter la méthode `main()` pour tester les méthodes ci-haut.

Question 3: **la classe `GestionEntrepot`**

La classe `GestionEntrepot` permet de gérer un entrepôt composé d'une liste de produits de type `ListeProduits`. Elle comporte des méthodes permettant à un utilisateur de saisir des informations à partir du clavier pour effectuer des opérations telles que l'ajout, la suppression et la recherche de produits dans la liste.

Dans la méthode `main()` de cette classe, une liste de 100 produits a été créée.

```
// créer un entrepot de 100 produits maximum
ListeProduits listeProd = new ListeProduits(100);
```

Travail à faire :
Complétez les méthodes suivantes:

`1. public static void initialiserEntrepot(ListeProduits listeProd)`
Cette méthode ajoute les 3 produits suivants dans `listeProd`:

```
100 Table 5000
150 Chaise 60
200  lit 500
```

Après écriture du code de cette méthode, exécutez la méthode `main()`, où elle est appelée, pour la tester.

`2. public static void ajouterProduit(ListeProduits listeProd)`
Cette méthode demande à l'utilisateur de rentrer les informations d'un produit à partir du clavier, elle initialise un objet produit avec ces informations qu'elle ajoute dans la liste `listeProd`.
Les grandes étapes du programme sont:

- Créez un objet de la classe `Produit` soit `prod` en appelant le constructeur par défaut .
- Utilisez `prod.lireNumero()`, `prod.lireDescription()` et `prod.lirePrix()` pour initialiser les attributs de `prod` à partir des informations lues du clavier.
- Appelez la méthode `ajouter()` de la classe `listeProduit` avec le bon argument pour ajouter le produit `prod` dans la liste `listeProd`. Si l'ajout est effectué, affichez le message `«Le produit a été ajouté avec succès»`, sinon affichez le message `«Échec d'ajout du produit»`

Voici un exemple de trace d'exécution:

```
Entrez le numéro du produit: 250
Entrez la description du produit: Sofa
Entrez le prix du produit: 368
Le produit a été ajouté avec succès.
```

Ajoutez des instructions dans la méthode `main()` permettant de tester cette méthode.

`3. public static void supprimerProduit(ListeProduits listeProd)`
Cette méthode demande à l'utilisateur de rentrer le numéro du produit à supprimer, puis supprime le produit correspondant à ce numéro s'il existe dans la liste `listeProd`, sinon elle affiche un message d'erreur.
Appelez la méthode `supprimer()` de la classe `ListeProduit`.

Voici 2 exemples de traces d'exécution :

Trace1

```
Entrez le numero du produit à supprimer:150
Le produit a ete supprime.
```

Trace 2

```
Entrez le numero du produit à supprimer:789
Le produit est introuvable.
```

Ajoutez des instructions dans la méthode main() permettant tester cette méthode .

4. `public static void rechercherPoduit(ListeProduits listeProd)`
Cette méthode demande à l'utilisateur de rentrer le numéro du produit à rechercher dans la liste `listeProd` à partir du clavier, puis affiche le produit s'il existe, sinon elle affiche un message de non-disponibilité du produit.
Les grandes étapes du programme sont:

a. Trouver l'indice du produit dont le numéro est rentré par l'utilisateur en appelant la fonction `trouverProduit()` de la classe `ListeProduits`, soit `indiceTrouve`.

b. Si `indiceTrouve` est différent de -1 (le produit existe), appelez la fonction `obtenirProduit()` de la classe `ListeProduits` en fournissant cet indice en argument pour avoir le produit, puis afficher ce produit en appelant `toString()` de la classe `Produit`. Sinon afficher le message d'erreur «`Le produit est introuvable`».

Voici 2 exemples de traces d'exécution :

Trace 1

```
Entrez le numéro du produit à rechercher: 100
Produit trouvé:
100 Table 500,00$
```

Trace 2

```
Entrez le numéro du produit à rechercher: 777
Le produit est introuvable
```

Ajoutez des instructions dans la méthode `main()` permettant tester cette méthode.

5. `public static void main(String[] args)`

Complétez la méthode `main()` si ce n'est pas déjà fait. Voici un exemple de trace d'exécution:

```
Initialise l'entrepôt
Les produits de l'entrepôt
100 Table           500,00$
150 Chaise          60,00$
200 lit             500,00$

Ajout d'un produit
Entrez le numéro du produit: 300
Entrez la description du produit: Armoire
Entrez le prix du produit: 1000
Le produit a été ajouté

Les produits de l'entrepôt
100 Table           500,00$
150 Chaise          60,00$
200 lit             500,00$
300 Armoire         1000,00$

Supprimer un produit
Entrez le numéro du produit à supprimer: 150
Le produit a été supprimé

Les produits de l'entrepôt
100 Table           500,00$
200 lit             500,00$
300 Armoire         1000,00$

Recherche de produits
Entrez le numéro du produit à rechercher: 200
Voici le produit que vous cherchez:
200 lit             500,00$
```