TP 3 : Tests

Exercice 1 : Stylos

L'objectif de cet exercice est d'implanter les trois classes `Pen`, `Cartridge` et `Cap` (exercice sur les stylos vu en TD2) et de valider votre implantation grâce aux fichiers de tests fournis sur Updago. La difficulté principale étant de gérer l'association bidirectionnelle entre le stylo et la cartouche. Contrairement à la version vue en TD, on ne fait aucune hypothèse ici sur la provenance de la cartouche lorsque l'on remplace la cartouche d'un stylo.

- Créez un nouveau projet `pen`.
- Ajoutez-y les 3 fichier `Pen.java`, `Cartridge.java` et `Cap.java` disponibles sur UPdago.
- Faites un clic droit sur le projet, puis `Tools > Create / Update Test`. Vous devez à présent voir apparaître un nouveau dossier `Test Packages` contenant un package `pen` avec 4 fichiers de tests.
- Remplacez les 4 fichiers de test par ceux fournis sur UPdago.
- Commencez par corriger les 3 classes (vides) afin qu'elles compilent.
- Vous pouvez à présent lancer l'ensemble des tests de chaque classe en faisant un clic droit sur la classe `PenITSuite`, puis `Test File`.
- Corrigez vos trois classes jusqu'à ce que tous les tests passent...
- Si vous le souhaitez, n'hésitez pas à ajouter de nouveaux tests.

Compléments d'informations sur les classes :

- L'appel de `freePen()` à partir d'une cartouche `cart` permet de "libérer" cart : elle ne sera plus liée à aucun stylo (autrement dit, son stylo vaut null). Évidemment, si `cart` était associée à un stylo `pen` avant l'appel de `cart.freePen()`, alors `pen` ne possèdera plus de cartouche après.
- La méthode `freeCart()` de `Pen` fonctionne de la même manière mais à partir d'un stylo.
- `cart.setPen(Pen pen)` permet de mettre la cartouche `cart` dans un stylo `pen`. Bien entendu, il faut mettre à jour tous les liens entre les différents stylos et cartouches mis en jeu. En particulier, si `cart` était déjà dans un stylo, alors il faut l'ôter ; et si `pen` possédait déjà une cartouche, alors il faut ôter cette cartouche.
- La méthode `changeCartridge(Cartridge c)` fonctionne de la même manière, mais à partir d'un stylo.
- La méthode `use()` diminue de 1 unité le niveau d'encre de la cartouche si celui-ci est strictement positif.
- La méthode `write(String)` permet d'écrire si le crayon possède une cartouche non vide. Cette méthode retournera `true` si le stylo a pu écrire, et `false` sinon.

Exercice 2 : Des portes

Reprenez l'exercice du TP2 sur les portes et écrivez des tests en vous inspirant de l'exercice précédent. Prenez le temps de réfléchir à tous les cas qui pourraient poser problème. Testez ensuite votre code et corrigez-le si nécessaire. Lorsque vous pensez que votre code est prêt à passer tous les tests qu'on lui soumettra, échangez vos tests avec ceux d'un(e) camarade et vérifiez si votre code est aussi fiable que prévu (si besoin, corrigez votre code à nouveau...).