FormuL∞

POGL TP6 : Tests avec JUnit

Dans le jeu *FormuL∞*, vous guidez un bolide (classe `Bolide`) autour d'un circuit de course (classe `Circuit`), en lui indiquant à chaque tour dans quelle direction accélerer. Dans ce TP, nous vous fournissons la classe principale `FormuLInf`, les squelettes des trois classes `Vect`, `Bolide` et `Circuit` que vous devez compléter, et une classe auxiliaire `Case`.

`http://www.lri.fr/~blsk/POGL/TDTP/FormuLInf.zip`

Petit rappel de physique. Une accélération est une modification de la vitesse, et une vitesse est donnée par un vecteur (classe `Vect`). Dans notre version nous considérons uniquement des positions et des vecteurs à coordonnées entières dans un espace à deux dimensions. Nous restreignons de plus l'intensité des accélérations, de sorte qu'une accélération (x, y) aura toujours une norme L∞ égale à 0 ou 1 : les coordonnées x et y ne pourront valoir que 0, 1 ou -1.

Nous ne dirons rien de plus sur les règles du jeu. En revanche, nous vous fournissons un jeu de tests unitaires pour chacune des méthodes à compléter, qui vous montrent les comportements attendus et qui devront vous guider dans l'écriture du code. Vous pouvez également obtenir à l'adresse

`http://www.lri.fr/~blsk/POGL/FormuLInf.jar`

une archive `.jar` que vous pouvez exécuter avec la ligne de commande

```
java -jar FormuLInf.jar mc1.flinf
```

pour tester le jeu vous-même sur le circuit exemple décrit dans le fichier `mc1.flinf` et en apprendre plus sur son fonctionnement.

Votre objectif : déduire des différents tests les spécifications de chaque méthode du squelette, et compléter le code de sorte que tous les tests unitaires réussissent et que votre jeu se comporte comme la version de référence.

Indications. Nous vous suggérons de compléter les classes dans l'ordre `Vect`, `Bolide`, `Circuit`, en traitant pour chacune les méthodes dans l'ordre dans lequel elles sont présentées. Vous pouvez également commencer sans vous soucier des cases affichées en vert (ces cases indiquent l'historique du tour précédent lorsque celui-ci est utile, et aucun des tests unitaires fournis ne s'y réfère).

À propos de JUnit 4. JUnit est un outil permettant d'écrire et d'exécuter des tests unitaires sur des programmes Java. Il est intégré à Eclipse mais est égalenemt disponible à l'adresse `http://www.junit.org/`.

Un test en JUnit 4 est une méthode annotée par `@Test`. Les méthodes de test sont généralement regroupées en une classe dédiée aux tests. Le corps d'une méthode de test doit comporter quatre parties :

- le *préambule*, qui permet de créer les objets et de les amener dans l'état nécessaire pour le test ;
- le *corps de test*, dans lequel la méthode à tester est appelée sur les objets créés ;
- l'*identification*, qui permet de délivrer le verdict du test (succès ou échec) en vérifiant un ensemble de propriétés (assertions) sur l'état des objets après le test.
- le *postambule*, qui réinitialise les objets.

Il est possible de grouper les tests ayant un préambule commun (c'est-à-dire devant être exécutés dans le même état) en une classe et de définir une méthode qui exécutera ce préambule avant chacun des tests de la classe. Cette méthode doit être annotée par `@Before`. De la même manière, si tous les tests d'une classe ont un postambule commun, on peut définir une méthode annotée par `@After` qui sera exécutée après chacun des tests de la classe.

Mise en route. Sous Eclipse, créez un nouveau projet java, et importez-y les fichiers du répertoire `FormuLInf/` fourni : clic droit sur le package `default` > `Import` > `File System` puis dans le répertoire `FormuLInf`, importez les trois sous-dossiers `formulinf`, `ig` et `test`. Ajoutez ensuite JUnit 4 au *classpath* : clic droit sur le projet > `Build Path` > `Add Librairies` > `Junit4`.