Tests en isolation avec JMock

Ce TP a pour but de vous familiariser avec le framework JMock permettant d'utiliser des simulacres dans vos tests. Commencez par récupérer le framework JMock (un fichier `.jar` qu'il vous suffira d'ajouter à votre chemin de compilation dans votre projet Eclipse). Vous pouvez consulter le support de cours disponible sur la liste de diffusion pour un (tout petit) exemple d'utilisation de simulacre à l'aide JMock.

Vous utiliserez la méthode du « TDD » pour réaliser ce TP. C'est donc volontairement que les composants qui vont suivre ont été sous-spécifiés.

1 Un jeu de dés

On s'intéresse à un jeu de dés impliquant une banque et des joueurs.

- Chaque joueur possède une somme d'argent qui lui permet de jouer à un jeu avec une certaine mise.
- Le jeu qui nous intéresse se joue avec 2 dés et une banque qui gère les pertes et les gains des joueurs.
- Les dés sont du modèle classique à tirage aléatoire entre 1 et 6. La banque possède un fond et un fond minimum à ne pas dépasser. Si elle passe en dessous de ce niveau suite à un pari gagnant, le gain est quand même donné aux joueurs, mais la banque saute et le jeu ferme immédiatement.
- La règle du jeu est la suivante :
  1. on ne peut jouer qu'à un jeu ouvert ;
  2. les joueurs qui ont pariés sont débités du montant de leur pari qui est encaissé par la banque ;
  3. ensuite les 2 dés sont lancés ;
  4. si la somme des lancers vaut 7 alors les joueurs gagnent : la banque paye deux fois la mise, somme créditée aux joueurs ;
  5. si le pari a fait sauter la banque, le jeu ferme immédiatement ;
  6. si la somme des lancers ne vaut pas 7 les joueurs ont perdu leur mise.

2 Sujet

Le sujet consiste à tester en isolation la méthode

```
public void jouer(int mise) throws ...
```

de la classe qui représente le jeu (paramètres de la méthode à adapter si besoin).

Le test sera donc purement unitaire au niveau de la classe. On ne demande pas d'écrire ni de tester les autres classes : limitez-vous à des interfaces et utilisez des doublures.

On ne demande pas non plus d'écrire les méthodes de la classe représentant le jeu qui ne seraient pas utilisées par jouer (argument désagréable : tout code non demandé vaudra des points en moins à la correction).

Les mocks devront rester les plus simples possibles : on peut lancer un dé, créditer ou débiter un joueur, créditer ou débiter une banque (avec levée d'exception si la banque saute en cas de débit).

- Commencer par le scénario le plus simple : le jeu est fermé.
- Écrire le test (il ne doit y avoir aucune interaction avec les autres acteurs).
- Ajouter les autres scénarios du même genre si vous en voyez.
- Continuer par le scénario d'un joueur perdant : un test d'interaction est nécessaire. Vérifier notamment que le joueur a été débité de sa mise, laquelle a été créditée à la banque, et que le jeu reste ouvert.
- Continuer avec les scénarios qui manquent.