The bowling project

Franck Silvestre <franck.silvestre@irit.fr>

Table of Contents

1. Introduction

Objectifs du projet

- Appliquer certaines pratiques de travail collaboratif avec Git, Github et l'IDE
- Mettre en oeuvre et construire un projet avec Maven
- Tester son code avec JUnit en étant guidé par la couverture du code par les tests
- Mettre en oeuvre des outils d'analyse statique de code pour contrôler et maintenir son code conforme aux bonnes pratiques de développement
- Tirer partie des rapports produit via Maven et ses plugins
- Préparer l'évaluation :-)

Organisation

- Travail en binôme
- 2 séances de TP + ...
- Déclaration du binôme sur sondage

Important

1 inscription par binôme

https://docs.google.com/forms/d/1yRDwkyO-0STaTIobDli7oBUeLQRwM6ha9zqxlNPLiWM/viewform?embedded=true

Évaluation

- 1 séance de TP de 3 heures
- Travail individuel
- Évaluation sur utilisation de Git
  - Clone d'un projet sur Github
  - Modifications de fichier triviales, commit, création de branches, merge, etc.
  - En cours de TP, push du code sur le projet Github
- Évalution sur utilisation de Maven
  - Création d'un projet Maven "quick start"
  - Installation de plugins d'analyse statique de code
  - Génération de rapport
- Évaluation sur utilisation de JUnit/Checkstyle/findbugs
  - Écriture des tests sur un code existant pour obtenir
    - 100% de couverture de code
    - Aucune règle Checkstyle/Findbugs violée
  - Modification du code principal suite à un jeu de test provoquant une erreur.

2. Exigences non fonctionnelles

Git, Github et IDE

- Utilisation de Git comme VCS
  - Création d'une branche par feature et/ou par contributeur
- Utilisation de Gihub pour le repository partagé
  - Utilisation d'un repository public
  - Utilisation des issues Github pour tracer les tâches exécutées par chaque développeur
- Lier commits et tâches déclarées dans Github
- Travailler en utilisant le gestionnaire de tâches de l'IDE

Mise en oeuvre de Maven

- Création du projet avec archetype Maven quickstart (vue en TP)
- Utilisation des plugins FindBugs, Checkstyle et Cobertura permettant l'analyse statique du code (vue en TP)
- Utilisation d'un JDK version 1.7xxx

Les tests

- Utilisation de JUnit 4.12
- Utilisation du plugin Maven Cobertura pour la mesure de la couverture
- 100% de couverture de code visée

L'analyse statique de code

- Findbugs: 0 bug, 0 error, 0 missing class
- Checkstyle: 0 violation de règle

3. Exigences fonctionnelles

Calcul du score d'une partie de bowling

Description du problème

Créer un programme qui calcule le score total d'une partie de bowling pour un joueur à partir de la séquence valide de lancers pour ce joueur.

Quelques éléments sur le programme :

- le programme vérifiera la validité de la séquence de lancers, le nombre correct de lancers et de jeux.
- Le programme ne calculera pas les scores intermédiaires

Synthèse des règles permettant le calcul du score

- Chaque partie de bowling est composée de 10 jeux (frames) pour un joueur.
- Dans chaque jeu, le joueur dispose de deux essais pour faire tomber toutes les quilles.
- Si en deux essais, le joueur ne réussit pas à faire tomber toutes les quilles, le score pour ce jeu est le nombre total de quilles tombées lors de ses deux essais.
- Si en deux essais, le joueur fait tomber toutes les quilles, on dit que le joueur a réalisé un "spare" et son score pour le jeu courant est de 10 plus le nombre de quilles tombées lors du prochain lancer (dans le prochain jeu).
- Si lors du premier lancer dans le jeu, le joueur fait tomber toutes les quilles, on dit que le joueur a réalisé un "strike" ; le jeu en cours est terminé et le score sur ce jeu est de 10 plus le nombre total de quilles tombées lors deux prochains lancers.

- Si le joueur réalise un spare ou un strike sur le dernier jeu, il bénéfice respectivement d'un ou deux lancers complémentaires afin de pouvoir comptabiliser les points du dernier jeu. Si durant le ou les lancers complémentaires, le joueur fait tomber toutes les quilles, le processus n'est pas reconduit, la partie est terminée pour le joueur.
- Le score de la partie est la somme des scores de chaque jeu.

Point d'attention

- Le choix de la structure utilisée pour décrire une séquence de lancers est libre.
- Ce choix est déterminant dans l'algorithme que vous mettrez en place.
- Il n'y a pas une unique solution.

Suggestions de cas de tests

Dans la séquence "X" indique un strike, "/" indique un spare, "_" indique qu'aucune quille n'est tombée.

- "XXXXXXXXXXXX" : 12 strikes
  => score = 300
- "9_9_9_9_9_9_9_9_9_9_" : à chaque jeu 1 lancer à 9, 1 lancer à 0
  => score = 90
- "5/5/5/5/5/5/5/5/5/5/5" : 10 spare où 5 quilles tombent à chaque premier lancer
  => score = 150

D'autres tests auront sans doute à être écrits.

Important

Le format utilisé dans ces cas de tests pour la description d'une séquence de lancers n'est absolument pas à considérer comme une indication du format à utiliser dans votre programme.