Les tests

Cyrille François © Ateno Tech

Table des matières

Introduction

- **Introduction sur les tests**
- **Tests Unitaires**
- TDD
- TI et Intégration continue

Tests Unitaires

1. Exercice : Tests

Le problème des tours de Hanoï est un jeu de réflexion consistant à déplacer des disques de diamètres différents d'une tour de **départ** à une tour d'**arrivée** en passant par une tour **intermédiaire** et ceci en un minimum de coups, tout en respectant les règles suivantes :

- On ne peut déplacer plus d'un disque à la fois
- On ne peut placer un disque que sur un autre disque plus grand que lui ou sur un emplacement vide

On suppose que cette dernière règle est également respectée dans la configuration de départ (où tous les disques se situent sur la tour de **départ**).

Le nombre de disques est fonction de la hauteur des tours.

La hauteur des tours, le nombre et le diamètre des disques seront paramétrables dans notre implémentation...

Le code du jeux est disponible sur github : https://github.com/CfrancCyrille/hanoi

Test partitionnel et analyse des valeurs limites (tests fonctionnels)

Nous avons créé une méthode *empiler* dans la classe Tour, pour empiler des disques définit par la classe **Disque** dont voici un extrait :

```
1 public class Disque implements Comparable<Disque> {
2   int d;
3   public Disque(int d) {
4       this.d = d;
5   }
6   int compareTo(Disque obj){
7     //[...]
8   }
9 }
```

La fonction *empiler* est déclarée dans l'interface **IPile** et est implémentée dans la classe **Tour** dont voici un extrait :

```
1 public class Tour implements IPile<Disque>{
2   protected int diam(){ //Retourne le diametre de l'élément situé au sommet
3     //[...]
4   }
5   public boolean empiler(Disque d){
6     //[...]
7   }
8   //[...]
9 }
```

Sélection de tests boîte blanche (tests structurels)

Soit le programme en langage Java suivant pour la classe **Tour** des Tours de Hanoï :

```java
public class Tour {

    Queue<Disque> disques=new ArrayDeque<Disque>();

    public int diam(){
       return this.disques.element().d;
    }

    public int taille() {
       return disques.size();
    }

    boolean empiler(Disque d){
        boolean res=false; // B1
        if(this.disques.isEmpty()){ // P1
            this.disques.offer(d); // B2
            res=true; // I1
        }
        else{
            if( (diam()>d.d) && (taille()<hauteurMax) ){{ // P2
                this.disques.offer(d); // B3
                res=true; // I2
            }
            else{
                res=false; // B4
            }
        }
        return res; // B5
    }
}
```

Tests unitaires en Java avec JUnit

Dans le cadre du jeu des des tours de Hanoï réalisé au TP1 (en Java)

Question 1

Compléter la classe de test **TourTest** (en JUnit) pour tester la méthode empiler de la classe Tour, en considérant les données de test recensés plus haut (tests fonctionnels).

Question 2

Compléter la classe de test **DisqueTest** (en JUnit) pour tester la méthode *compareTo* de la classe Disque.

Question 3

Compléter la classe Hanoi pour résoudre le jeux avec :

- 3 disques
- n disques (nombre paramétrable de disques)

TDD

II

1. Exercice : Kata Roman number

Kata les nombres romains

Écrire une méthode de conversion d'un nombre "normal" en nombre romain

Nombre	Symbole
1	I
5	V
10	X
50	L
100	C
500	D
1000	M

Les nombres romains

Plus d'informations sur les nombres romains :

http://www.novaroma.org/via_romana/numbers.html

Kata original : *http://codingdojo.org/kata/RomanNumerals/*

Question 1

Tester :

1 : I

2 : II

3 : III

4 : IV

5 : V

9 : IX

10 : X

Question 2

Passer à des nombres de 11 à 90