Compte Rendu Projet JML (TD 4)

Maxime Dapp, Jérôme Ferrafiat

Février 2018

Question 1 - lecture et test d'invariant

Commentaire sur les propriétés:

prop 1: Il ne peut pas y avoir moins de 0 incompatibilités, et il ne peut y en avoir plus de 49.

prop 2: Il ne peut pas y avoir moins de 0 assignations de produits, et il ne peut y en avoir plus de 29.

prop 3: Pour toute ligne l du tableau des incompatibilité, l[0] et l[1] sont des produits.

prop 4: Pour toute ligne l du tableau d'assignations, l[0] est un bâtiment et l[1] est un produit.

prop 5: Un produit ne peut pas être incompaptible avec lui-même.

prop 6: Cette propriété sert à vérifier le caractère symétrique de la relation d'incompatibilité. Si a est incompatible avec b, alors b est incompatible avec a.

prop 7: Cette propriété vérifie qu'aucun bâtiment ne contiennent de produits incompatibles entre eux.
Les tests sont dans les fichiers TestExplosivesJUnit4.java et TestExplosivesJUnit4Public.java

Question 2 - calcul de préconditions

fonction add_incomp :

le nombre d'incompatibilité doit être strictement inférieur à 48 car la fonction l'incrémente de 2.
Le nombre d'incompatibilité doit être supérieur ou égal à 0.
On sait aussi que les deux produits ne doivent pas être identiques.
Les deux produits doivent être des produits.

On ajoute donc les ligne:

```
//@ requires nb_inc < 48;
//@ requires nb_inc >= 0;
//@ requires !prod1.equals(prod2);
//@ requires prod1.startsWith("Prod") && prod2.startsWith("Prod");
```

fonction add_assign :

le nombre d'assignations doit être strictement inférieur à 29 car la fonction l'incrémente de 1.
Le nombre d'assignation doit être supérieur ou égal à 0.
bat doit être un batiment et prod un produit.

On ajoute donc les ligne:

```
//@ requires nb_assign < 29;
//@ requires nb_assign  >= 0;
//@ requires prod.startsWith("Prod") && bat.startsWith("Bat");
```

Question 3 - Ajout d'insertions

Les insertions 8 et 9 sont ajouté en tant que public invariant.
Pour l'insertion 10, on peut se baser sur un @ensure.

Question 4 - Recherche d'un bâtiment

La solution évoqué dans l'énoncé, qui s'avère être juste, bien que complétement inéfficace serait une version gloutonne, qui pour chaque batiments, vérifie qu'il n'y a pas d'incompatibilité et renvoie le premier élligible.