MGL7160 Hiver 2014
Date de remise : le 29 avril 2014

Projet partie III

– L'objectif de ce travail est de réaliser des contrats JML, de les analyser à l'aide de l'outil KeY ainsi que d'interpréter les résultats obtenus. Plus précisément, voici ce que vous devez faire.

1. Au sujet du contrat

```
// TP3 Q1.
     public normal_behavior
     requires article.\inv;
     requires poids + article.poids > poidsMax;
     ensures \result == false;
```

de la classe `Camion`, répondez aux question suivantes.

(a) Expliquer le sens de ce contrat en détaillant le sens exact de chacune des lignes.

(b) Est-ce que la clause `requires article.\inv;` est nécessaire à la vérification ? Pouvez-vous expliquer pourquoi, en justifiant votre réponse en fonction des séquents produits par l'outil ?

2. Assurez-vous d'avoir bien les valeurs par défaut pour la stratégie de preuve et que la méthode de traitement des méthodes est bien `Contract`. Ceci veut dire que plutôt que de substituer le code des méthodes appelées, le système utilisera les contrats de ces méthodes comme hypothèses.

(a) Pour le contrat précédent, pouvez-vous déterminer quel contrat de quelle méthode est pris comme hypothèse ?

(b) Justifiez que le système n'utilise pas le code de cette méthode comme hypothèse, mais seulement son contrat. Vous pouvez réaliser ceci en modifiant le code vers quelque chose d'erroné, mais en montrant que le système n'en tient pas compte.

(c) Pouvez-vous décrire une méthodologie de vérification (quelles méthodes doivent être vérifiées et dans quel ordre ?) adéquate pour cet usage des contrats ? Par exemple, pour la question précédente comment et à quel moment pourrait-on vérifier que le code de la méthode appelée est incorrect ?

3. Toujours pour la méthode `charger` de la classe `Camion`, donnez un contrat qui exprime que sous les bonnes conditions on aura qu'après l'exécution le `poids` du `Camion` sera son ancien `poids` plus celui de l'article.

Vous devez bien observer les séquents ouverts ainsi que le nom des règles de branchement pour détecter tout ce qui pourrait manquer comme préconditions. A priori, le système ne peut pas deviner les contraintes entre les attributs, mais s'il manque quelque chose la contrainte devrait se retrouver dans le séquents ouvert ou sinon implicitement dans une des règles de branchement.

Faites particulièrement attention aux points suivants.

(a) Il est nécessaire d'indiquer `requires \typeof(livraisons) == \type(Livraison[]);`, pour s'assurer que le type soit un tableau de `Livraison` et non un tableau d'un de ses sous-types. Pouvez-vous expliquer pourquoi ?

(b) Il est indispensable d'indiquer toutes les contraintes sur `prochaineLivraison`.

(c) Il faut mettre une contrainte sur `livraisons.length`, car a priori cette valeur peut être quelconque.

4. Dans le contrat que vous avez donné à la question précédente, quelles parties pourraient être déplacées dans un invariant pour la classe ? Justifiez votre choix et pourquoi rien de plus ne peut être déplacé dans l'invariant. Réalisez votre solution à l'aide de l'outil pour montrer que votre raisonnement est juste. Observez l'invariant de la classe `Article` pour vérifier la syntaxe exacte.

– **Ce que vous devez remettre :**
  – Dans un document papier vous devez répondre à toutes les questions autant celles d'analyse que celles demandant de rédiger des contrats JML.
  – Vous devez aussi exporter vos preuves avec `File -> Save`, les joindre au code source complet incluant vos contrats JML et mettre le tout dans une archive `zip`.
  – **Faites moi parvenir le tout à l'adresse villemaire.roger@uqam.ca** au plus tard à la date de remise.