Institut für Theoretische Informatik
Logik und Formale Methoden
Prof. Dr. P. H. Schmitt, C. Engel, D. Faragó, R. Krenický, M. Ulbrich, B. Weiß

25.01.2010

Praktikum
Formale Entwicklung objektorientierter Software
Übungsblatt 9: KeY – Gruppe 3

Aufgabe 22 — Milliardenbank

In dieser Aufgabe sollen Sie den Vertrag der Methode `bank.Account.addTransaction` des Praktikumsprojekts verifizieren. Um den Code in KeY zu laden, müssen Sie das Verzeichnis `bank/` Ihrer Implementierung mit KeY öffnen.

Schreiben Sie, falls nicht schon geschehen, einen Methodenvertrag für `addTransaction(t)`, der insbesondere sicherstellt, daß

- die Transaktion `t` dem Transaktionsarray hinzugefügt wurde,
- die anderen Einträge im Transaktionsarray nicht verändert wurden.

Benutzen Sie für den Vertrag außerdem folgende `assignable`-Klausel:

```
assignable transactions, transactions[amountOfTransactions],
           amountOfTransactions, \object_creation(Transaction[]);
```

Bevor Sie die Methode `addTransaction` beweisen können, werden Sie noch einige "Hilfsspezifikationen" schreiben müssen: Die Implementierung von `addTransaction` ruft die Methode `resizeTransactions` auf. Im Sinne einer modularen Verifikation wollen wir Methodenaufrufe nur mit Hilfe der Spezifikation der aufgerufenen Methode auswerten (wie es im übrigen auch *ESC/Java2* tut). Wir benötigen also zunächst eine Spezifikation für `resizeTransactions`, die stark genug ist, damit sich daraus die Korrektheit der gegebenen Spezifikation für `addTransaction` folgern läßt.

Um wiederum die Korrektheit des (von Ihnen zu schreibenden) Kontrakts von `resizeTransactions` zu beweisen, wird außerdem eine Schleifeninvariante (inkl. Variante und Assignable-Klausel) benötigt.

- Die Klasse `Account` sollte folgende Invarianten enthalten:
  - `invariant \typeof(transactions) == \type(Transaction[]);`
  - `invariant (\forall int i; 0 <= i && i <amountOfTransactions; transactions[i] != null);`

  Möglicherweise müssen Sie weitere sinnvolle Invarianten schreiben.
- Fügen Sie `resizeTransactions` die oben beschriebenen Spezifikationen (Methodenvertrag und Schleifeninvariante) hinzu, und beweisen Sie die Korrektheit des Methonvertrags mit Hilfe der Beweisverpflichtungen EnsuresPost (Korrektheit der Nachbedingung) und RespectsModifies (Korrektheit der Assignable-Klausel).
- Stellen Sie dabei sicher, daß folgende Strategieoptionen bei der Strategie *Java DL* eingestellt sind:
  - Goal Chooser: Depth First

  - Stop at: Default
  - Loop treatment: Invariant
  - Method t.: Contracts
  - Query t.: None
  - Arithmetic t.: DefOps
  - Quantifier t.: No Splits with Progs

- Sobald Sie es geschafft haben, einen korrekten (und hinreichend starken) Vertrag für `resizeTransactions` zu formulieren, beweisen Sie mit dessen Hilfe den Vertrag von `addTransaction` (wieder: EnsuresPost, RespectsModifies).

- Alle bisherigen Beweise sollten (bei geschickter Wahl der Spezifikationen) ohne Benutzerinteraktion zu Schließen sein.

- Beweisen Sie außerdem, daß `resizeTransactions` die Invarianten der Klasse `Account` erhält (Beweisverpflichtung: PreservesOwnInv). Diesmal gelingt es der automatischen Beweisstragie nicht, den Beweis selbständig zu finden, daher müssen Sie einige Quantoren interaktiv instanziieren. Welche das sind und wie die Instanziierungen zu wählen sind, läßt sich anhand der offenen Beweisäste herausfinden.

Abgabe bis Mittwoch, 03.02.
Die Abgabe erfolgt über die Praktikums-Webseite. Es braucht pro Gruppe und Aufgabe nur *eine* Lösung abgegeben werden. Bitte dokumentieren Sie Ihre Lösungen ausreichend und seien Sie darauf vorbereitet, sie auf Nachfrage zu erklären.

Praktikums-Webseite: `http://lfm.iti.kit.edu/keyprakt0910.php`

Christian Engel:	R. 106,	Tel. 608-4338,	E-Mail: `engelc@ira.uka.de`
David Faragó:	R. 308,	Tel. 608-7322,	E-Mail: `farago@ira.uka.de`
Roman Krenický:			E-Mail: `krenicky@ira.uka.de`
Mattias Ulbrich:	R. 106,	Tel. 608-4338,	E-Mail: `mulbrich@ira.uka.de`
Benjamin Weiß:	R. 309,	Tel. 608-4324,	E-Mail: `bweiss@ira.uka.de`