

Aufgabe 1 (H) (*Small-Step-Semantik von Jinja*)

Bestimmen Sie jeweils für den Ausdruck e den vollständig reduzierten Ausdruck e' in der Small-Step-Semantik. Dabei wird von einem beliebigen Heap h und den lokalen Variablen l ausgegangen. Geben Sie in Ihrer Lösung die vollständige Herleitung von

$$P \vdash \langle e, (h, l) \rangle \rightarrow^* \langle e', s' \rangle$$

an, bis e' nicht mehr weiter reduziert werden kann.

- a) $e = x := \text{Val } (\text{Bool True}); y := \text{Val } (\text{Bool False}); \text{Var } x$
- b) $l = (V \mapsto v_0)$
 $e = \{V : T; \{V' : T'; V := \text{Val } v; V' := \text{Var } V; \text{Var } V'\}\}$

Hinweis: Machen Sie sich bei b) zunächst anhand der Big-Step-Semantik klar, was e' und s' ist. Geben Sie jeweils in der Herleitung an, welche der Regeln für lokale Variable Sie benutzten. Hierzu seien die Regeln in der Reihenfolge, in der sie auf den Vorlesungsfolien eingeführt werden, mit Loc1 bis Loc5 bezeichnet.

Aufgabe 2 (Ü) (*Finale Ausdrücke und Exceptions*)

In der Vorlesung wurden für die Big-Step-Semantik von Jinja mit Exceptions finale Ausdrücke folgendermaßen definiert:

$$\text{final } e \equiv (\exists v. e = \text{Val } v) \vee (\exists a. e = \text{Throw } a)$$

Wie muss die Definition für *finals es* aussehen, so dass das folgende Lemma gilt:

Wenn $P \vdash \langle e, s \rangle \Rightarrow \langle e', s' \rangle$, dann *final* e' und
wenn $P \vdash \langle es, s \rangle [\Rightarrow] \langle es', s' \rangle$, dann *finals* es' .

Geben Sie einen Beweis für das Lemma mit Regelinduktion über die Relationen \Rightarrow und $[\Rightarrow]$ an.

Hinweis: zur Bestimmung von *finals es* genügt es, die Regeln auf der Folie “Exceptions: Method call” zu betrachten.

Aufgabe 3 (P) (*Deklarationsinformation in Prolog*)

In dieser und den Programmieraufgaben der folgenden Übungsblätter wird die Semantik von Jinja in Prolog implementiert. Als ersten Schritt hierzu implementieren Sie die Prädikate, mit denen Informationen aus einem Jinja-Programm extrahiert wird, insbesondere $P \vdash C \preceq^* D$, $P \vdash C \text{ has-fields } FDTs$, $P \vdash C \text{ sees } F : T \text{ in } D$, $P \vdash C \text{ has } F : T \text{ in } D$ und $P \vdash C \text{ sees } M : Ts \rightarrow T = B \text{ in } D$.

(Bitte wenden.)

Das Prädikat `has` wurde in der Vorlesung nicht eingeführt. $P \vdash C \text{ has } F : T \text{ in } D$ bedeutet, dass im Programm P eine (nicht notwendig echte) Oberklasse D von C ein Feld F vom Typ T hat, unabhängig davon, ob das Feld von C aus sichtbar ist, oder nicht.

Einen Programmrahmen finden Sie im Verzeichnis

```
/usr/proj/semantik/prog/prolog/jinja-declare
```

in der Datei `declare.pl`. Geben Sie Ihre modifizierte Datei ab.

Hinweise: Benutzen Sie für Maps Assoziationslisten und verwenden Sie das Prädikat `lookup`, das in `map.pl` vorgegeben ist, um einen Eintrag aus einer Map auszulesen. Das in `init.pl` deklarierte Prädikat `prog1` ermöglicht es Ihnen, Ihre Implementierung an dem Beispielprogramm aus der Vorlesung zu testen.