UFR-IMAG
Université Joseph Fourier

Programming Language and Compiler Design,
2009/2010

Marion Daubignard
Yassine Lakhnech
Laurent Mounier

Series 1

Exercise 1

Let consider the following statement $(z := x; x := y); y := z$, and the environment σ_0 which maps every variables but x and y to 0, maps x to 5, and y to 7. Give a derivation tree of this statement.

Exercise 2

We consider the arithmetical expressions defined in the lecture course.
Let $a, a' \in \mathbf{Aexp}$, and σ, σ' two states. Let X be the set of variables appearing in a.

1. Prove that if $\forall x \in X \cdot \sigma(x) = \sigma'(x)$, then $\mathcal{A}[a]\sigma = \mathcal{A}[a]\sigma'$.
2. Prove that $\mathcal{A}[a[a'/x]]\sigma = \mathcal{A}[a]\sigma[x \mapsto \mathcal{A}[a']\sigma]$.

Exercise 3

We consider the following statements:

- while $\neg(x = 1)$ do $(y := y * x;\ x := x - 1)$ od
- while $1 \leq x$ do $(y := y * x;\ x := x - 1)$ od
- while true do skip od

where x designates a variable of type $\mathbb{Z}$.
For each of the preceeding statement, determine whether:

1. its execution loops in every state
2. its execution stops in every state

Prove your answers.

Exercise 4

We wish to add the following statement to the **While** language:

repeat S until b

1. Provide the semantics rules in order to define repeat S until b without using the while b do $\cdots$ od construction.
2. Prove that
   (a) repeat S until b
   (b) S; if b then skip else (repeat S until b).
   are semantically equivalent.

3. We want to prove that the statement `repeat` S `until` b does not add any expressive power to the **While** language. To do so, give a function which transforms every program with the statement `repeat` S `until` b into a program in the **While** language. Is the given transformation computable? Compare the size of a program and the size of its image by this transformation.

Exercise 5

Prove that, for all statements S_1, S_2, S_3, the following statements are semantically equivalent:

1. $S_1; (S_2; S_3)$
2. $(S_1; S_2); S_3$

Prove that, in general, $S_1; S_2$ is not semantically equivalent to $S_2; S_1$.

Exercise 6

Prove that the natural operational semantics of the **While** language is deterministic.

Exercise 7

Give a natural operational semantics for arithmetical and boolean expressions which is equivalent to the inductive semantics given in the lecture course.