Homework # 3
due March 11, 13:00

Turn in your explanations (§3) on paper at the beginning of lecture, and send your SASyLF proofs (§2) as a file `lambda-eval.slf` to `scmalte@inf.ethz.ch` before 1 pm.

1 Reading

Please read Chapter 5 in your textbook.

2 Proofs

Prove the following theorems in SASyLF:

1. Prove that *multi*-step full beta-reduction of ω where

   $$\omega = (\lambda x \cdot x\, x)(\lambda x \cdot x\, x)$$

   never reaches a "value," (a lambda abstraction). To do this, you need to specify multi-step full beta-reduction. If you use the same rules for multi-step evaluation as in Homework #2, you will require another lemma. It is easier if you define multi-step evaluation to be right recursive.

2. Prove that call-by-value evaluation is deterministic.

3. Prove that every term either can evaluate one step (using call-by-value evaluation) or that it is a value (a lambda term).

3 Evaluation Orders

1. For each of the following situations, give a pure lambda-calculus term that has the given properties:

   (a) diverges under normal evaluation but not under call-by-value

   (b) diverges under call-by-value but not under normal evaluation

   (c) diverges under normal evaluation and under call-by-value evaluation, but not under call-by-name

   You may use ω defined above.

2. Call-by-need is used in lazy languages such as Haskell. Explain how it differs from the three: call-by-value, call-by-name and normal evaluation. Explain which of the three diverges on exactly the same terms as call-by-need, while still not being the same. Explain on paper.