Assignment 11
Semantics, WS 2011-2012

Prof. Dr. Gert Smolka, Dr. Chad Brown
www.ps.uni-saarland.de/courses/cl-ss11/

Read in Software Foundations:

Exercise 11.1 Write a function in T that adds two numbers. Translate your function to Coq and test it for some arguments. Note that T translates directly to Coq with *primrec* as defined above.

Exercise 11.2 Define an abstract syntax and a small-step semantics for T in Coq. Follow the development of PCF.

Exercises 11.3 - 11.7 are about the functional language E.

Exercise 11.3

a) Normal terms that are not values are called **stuck**. Find a stuck term.

b) Find an example showing that the step relation does not preserve types from right to left.

Exercise 11.4 Suppose we add two new reduction rules:

$$\begin{aligned} P\ true &\to P\ false \\ P\ false &\to P\ true \end{aligned}$$

Which of the following properties remain true in the presence of these rules?

a) Determinacy of *step*

b) Termination of *step* for well-typed terms

c) Progress

d) Preservation

Exercise 11.5 Suppose we add a new typing rule:

$$\frac{t_1 : T}{if\ \ true\ t_1\ t_2 : T}$$

Which of the following properties remain true in the presence of these rules?

a) Determinacy of *step*

b) Termination of *step* for well-typed terms

c) Progress

d) Preservation

Exercise 11.6 Prove the following lemmas.

```
Lemma value_normal t t' :
value t -> step t t' -> False.
```

```
Lemma preservation t T t' :
type t T -> step t t' -> type t' T.
```

```
Lemma progress t T :
type t T -> value t \/ exists t', step t t'.
```

```
Lemma type_unique t T T' :
type t T -> type t T' -> T = T'.
```

```
Lemma step_deterministic t t1 t2 :
step t t1 -> step t t2 -> t1 = t2.
```

Exercise 11.7 Prove that *step* terminates.

Exercise 11.8 Prove the following lemmas about the type-indexed version of E.

```
Lemma step_deterministic (T : ty) (t t1 t2 : tm T) :
step t t1 -> step t t2 -> t1 = t2.
```

```
Lemma Progress (T : ty) (t : tm T) :
value t \/ exists t', step t t'.
```

Exercise 11.9 Formalize STLC in Coq. For the abstract syntax and the small-step semantics of follow the development of PCF. For contexts and the typing relation follow the development in the SF text.

Exercise 11.10 (Optional) Reconsider the functional language E. Write a function *tycheck* : *tm* → *option ty* and prove the following lemma.

```
Lemma tycheck_correct t T :
type t T <-> tycheck t = Some T.
```