Assignment 2
Semantics, WS 2007/08

Prof. Dr. Gert Smolka, Dr. Jan Schwinghammer
www.ps.uni-sb.de/courses/sem-ws07/

Read Chapter 3 of **Types and Programming Languages**

We implement the abstract syntax of PCF as follows:

```
datatype ty = Bool | Int | Proc of ty * ty
type var = string
datatype ter =
   False | True | If of ter*ter*ter
 | 0 | Succ of ter | Pred of ter | Iszero of ter
 | V of var | A of ter*ter | L of var*ty*ter | Fix of ter
```

Exercise 2.1 (Addition) Declare an identifier *add* of type *ter* that represents the abstract syntax of a PCF function for addition. Use *add* to test your solutions to the following exercises.

Exercise 2.2 (Free variables) Write a procedure $free : var \to ter \to bool$ that checks whether a variable occurs free in a term.

Exercise 2.3 (Values) Write a procedure $isVal : ter \to bool$ that tests whether a term is a value.

Exercise 2.4 (Elaboration) Write a procedure $elab : (var \to ty) \to ter \to ty$ that yields the type of a term. Raise the exception *Error* if the term is not well-typed. The empty type environment and an update operation for environments can be declared as follows:

```
exception Error
fun empty x = raise Error
fun update f x t y = if y=x then t else f y
```

Exercise 2.5 (Substitution) Write a procedure $subst : ter \to var \to ter \to ter$ that yields $t[x := s]$ if s is a closed term.

Exercise 2.6 (Evaluation) Write a procedure $eval : ter \to ter$ that yields the value of a closed term if it exists. Raise the exception *Error* if *eval* must quit because of a type inconsistency or a free variable occurrence.