Assignment 4
Semantics, WS 2007/08

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

Read Chapter 2.7–2.9 of the lecture notes, and Chapter 5 of Pierce's **TAPL**

Exercise 4.1 (PCF contextual equivalence) Find contexts C that separate

a) *false* and *true*,

b) *0* and *true*,

c) *succ*(*succ 0*) and *succ*(*succ*(*succ 0*)),

d) *fix* $(\lambda f.0)$ and *fix* $(\lambda f.\lambda x.0\ x)$,

e) *if* x *then* $(f\ true)$ *else* $(f\ false)$ and $f\ x$.

Let us write $t \Uparrow$ if there is no v such that $t \Downarrow v$. Then, for all terms t and evaluation contexts E, $t \Uparrow$ implies $E[t] \Uparrow$.

f) Does this implication also hold for *arbitrary* contexts C, i.e., $t \Uparrow \implies C[t] \Uparrow$?

g) Does the converse hold, i.e., $E[t] \Uparrow \implies t \Uparrow$?

Exercise 4.2 (Delayed evaluation) Write declarations that represent the stream $x = x_0 :: x_1 :: \ldots$ where

$$\begin{aligned} x_0 &= 0 \\ x_1 &= 1 \\ x_{n+2} &= x_n + x_{n+1} + 1 \end{aligned}$$

a) with *lazy* in Alice ML;

b) with λ-lifting in Standard ML.

Exercise 4.3 (Untyped lambda calculus) Make sure that you can state the definitions of $\Downarrow$, $\to_0$, $\to$, $\rhd$, and $\sim$, adapted from PCF to the untyped λ calculus.

Exercise 4.4 (Church numerals) Make sure that you can give the definitions of *0*, *succ*, *plus*, *times*, *power*, and *iszero* in the untyped λ calculus.

Exercise 4.5 (Computing with Church numerals) Write the following procedures in the untyped λ calculus:

a) A procedure *pred* that yields $\max\{0, n-1\}$ for the number n. (More precisely, it yields a term equivalent to the Church numeral $c_{\max\{0,n-1\}}$ when applied to the n-th Church numeral c_n).

b) A procedure *fac* that yields $n!$ for n.

c) A procedure *minus* that yields $\max\{0, m - n\}$ for two numbers m, n.

d) A procedure *leq* such that *leq* c_n c_m yields *true* if $n \leq m$, and *false* otherwise.

Exercise 4.6 (Untyped lambda calculus in SML) This exercise asks you to implement in Standard ML an interpreter for the untyped λ calculus that can be used to experiment with Church arithmetic. The calculus has additional values

$$z \in \mathbb{Z}$$
$$v \in \mathit{Val} ::= \ldots \mid z \mid S$$

and one additional proper reduction rule:

$$\frac{}{S\ z \to z'}\ z' = z + 1$$

Represent λ terms as follows:

```
type var = string
datatype ter = V of var | L of var * ter | A of ter * ter
             | I of int | S
```

a) Write a procedure *eval* : *ter* $\to$ *ter* that evaluates terms according to $\Downarrow$.

b) Write a procedure *church* : *int* $\to$ *ter* that yields the Church numeral c_n when applied to $n \geq 0$.

c) Write a procedure *dechurch* : *ter* $\to$ *int* that yields n for terms that are equivalent to c_n.