Fachbereich Mathematik und Informatik
Prof. Dr. K. Ostermann

Sebastian Erdweg, seba@informatik
Tillmann Rendel, rendel@informatik

Philipps Universität Marburg

January 28, 2010

Programming Languages and Types
Homework Assignment 13

Please hand in your homework by email to `mailto:pllecture@informatik.uni-marburg.de` until February, 4. Please submit your solutions in appropriate file formats.

H13.1 Encoding Existential Types

Consider the following encoding of a counter with existentials.

```
counterADT = { *Nat,
               { new = 1,
                 get  = λi : Nat . i,
                 inc  = λi : Nat . succ(i) }}
            as { ∃Counter,
               { new : Counter,
                 get  : Counter → Nat,
                 inc  : Counter → Counter }}

test = let {Counter, counter} = counterADT in
       get (inc (inc new))
```

Rewrite this example using universal types. You can do this either in the formal System F notation, or you can write a little Haskell program using Rank-N types. In the latter case the type applications are implicit.

H13.2 Higher-Order Types

In F_ω, the universe of kinds can be separated into kind levels as follows.

$$K(1) = \{\}$$
$$K(i+1) = \{*\} \cup \{\kappa_1 \Rightarrow \kappa_2 \mid \kappa_1 \in K(i) \wedge \kappa_2 \in K(i+1)\}$$

For example, $* \Rightarrow *$ is in $K(3)$, $K(4)$ etc. but not in $K(2)$.
Corresponding to these levels, F_ω can be divided into sublanguages F_i, where the language F_i only permits kinds from kind level $K(i)$.
In this terminology, the simply-typed lambda calculus is F_1, and System F is F_2.
Write or find a useful program that can be written in F_4 but not in F_3.