Assignment 3
Introduction to Computational Logic, SS 2007

Prof. Dr. Gert Smolka, Mark Kaminski (MSc)
www.ps.uni-sb.de/courses/cl-ss07/

Read in the lecture notes: Chapter 2

This assignment asks you to implement some operations for terms in Standard ML. If you have not used ML before, we recommend you use the Alice interpreter (google for “Alice ML”). For an introduction to ML, google for “introduction to ml”.

Exercise 3.1 (About Abstractions) Answer the following questions. In case your answer is “yes”, give an example.

a) Does there exist an abstraction t and a name x such that there is no term s such that $t = \lambda x.s$?

b) Does there exist an abstraction t such that there is no name x and no term s such that $t = \lambda x.s$?

c) Does there exist a name x such that there are 2 different terms s and s' such that $\lambda x.s = \lambda x.s'$?

Exercise 3.2 (Subterms) For each of the following terms, give the set of all proper subterms.

a) $fx(gx)$

b) $\lambda x.x$

c) $(\lambda x.x)y$

Exercise 3.3 (Preterms)

a) Give 2 different preterms for the term $\lambda x.x$.

b) How many preterms exist for the term $\lambda x.x$?

c) How many preterms exist for the term $f(gx)$?

Exercise 3.4 (Canonical Preterms) Let a term structure with $Nam = \mathbb{N}$ be given and assume that the canonical name for an abstraction is the least name that does not occur in the abstraction. Then $L\,0\,0$ and $L\,0\,(L\,1\,(A\,0\,1))$ are canonical preterms that evaluate to the terms $\lambda x.x$ and $\lambda fx.fx$. Give the canonical preterms for the following terms.

a) $\lambda xy.y$

b) $\lambda xy.x$

c) $\lambda fgx.f(gx)$

Exercise 3.5 (Implementation of Terms) We implement terms in Standard ML as follows:

```
type nam = int  (* non-negative *)
datatype ter = N of nam | L of ter | A of ter * ter
```

a) Write an expression that yields the term $\lambda xy.fyx$ with $f = 7$.

b) Write a procedure $rep : nam \to ter \to nam \to ter$ that yields for x and t the substitution $[x := t]$.

c) Write a procedure $size : ter \to int$ that yields the size of a term.

d) Write a procedure $occurs : nam \to ter \to bool$ that tests whether a name occurs in a term.

e) Write a procedure $subst : (nam \to ter) \to ter \to ter$ that applies a substitution to a term.

f) Write a procedure $lam : nam \to ter \to ter$ that yields the abstraction $\lambda x.s$ for x and s.

Exercise 3.6 (Implementation of Reduction) Complete the following declarations such that they declare a procedure $red : ter \to ter$ that yields the $\beta\eta$-normal form of a term.

```
fun red (N x) = N x
  | red (A(s,t)) = beta (red s) (red t)
  | red (L t) = eta (red t)
and beta (L s) t = ··· apply β ···
  | beta s t = A(s,t)
and eta (A(t,N 0)) = ··· try η ···
  | eta t = L t
```

Exercise 3.7 (Implementation of Type Checking) We implement typed terms in Standard ML as follows:

```
datatype ty = S of string | F of ty * ty
type nam = int  (* non-negative *)
datatype ter = N of nam | L of ty * ter | A of ter * ter
```

Write a procedure $tau : ter \to ty$ that yields the type of well-typed terms that don't contain names. If a term contains a name or is not well-typed, tau should raise the exception *Domain* (use `raise Domain` to do this).

Exercise 3.8 (Schönfinkel's U) In his paper from 1924, Moses Schönfinkel showed that the Boolean operations and the quantifiers can be expressed with the function

$$U \in (X \to \mathbb{B}) \to (X \to \mathbb{B}) \to \mathbb{B}$$
$$U f g = (\forall x \in X \colon \neg(fx \wedge gx))$$

If f and g are interpreted as subsets of X, then U tests whether f and g are disjoint. Describe the following functions with U.

a) negation $\neg \in \mathbb{B} \to \mathbb{B}$

b) disjunction $\vee \in \mathbb{B} \to \mathbb{B} \to \mathbb{B}$

c) existential quantification $\exists \in (X \to \mathbb{B}) \to \mathbb{B}$