CS 565 Out: March 5, 1998
Jens Palsberg Due: March 19, 1998

Homework 7:

Implement Type Inference

Consider the language from Lecture Note 7. For each of the following λ-terms, write the constraint system. Decide if the λ-term has a simple type or not by doing unification of the constraints. If the λ-term has a simple type, then give a type derivation.

1. The S combinator:

$$\lambda x.\lambda y.\lambda z.(xz)(yz)$$

2. The twice function:

$$\lambda f.\lambda x.f(f(x))$$

3. A fun λ-term:

$$(\lambda z.(z\ \mathsf{0}))(\lambda y.\lambda x.x(\mathsf{succ}\ y))$$

4. A non-famous λ-term:

$$\lambda x.\lambda y.(y(x(\lambda a.a)))(x(\lambda z.z))$$

5. A not-at-all famous λ-term:

$$\lambda g.(\lambda f.(fg)f)(\lambda a.\mathsf{succ}\ a)$$

6. Another not-very-famous λ-term:

$$(\lambda f.((\lambda b.\lambda z.((bf)f)z)(\lambda x.\lambda y.xy)))(\lambda a.\lambda b.b)$$

7. Here is the grammar for a subset of Scheme:

$$
\begin{array}{rcl}
\textit{Exp} & ::= & \textit{VariableReference} \\
 & | & \texttt{(lambda (}\ \textit{Variable}\ \texttt{)}\ \textit{Exp}\ \texttt{)} \\
 & | & \texttt{(}\ \textit{Exp Exp}\ \texttt{)} \\
 & | & \textit{Constant} \\
 & | & \texttt{(}\ \textit{UnaryOp Exp}\ \texttt{)} \\
 & | & \texttt{(if}\ \textit{Exp Exp Exp}\ \texttt{)} \\
 & | & \texttt{(set!}\ \textit{Variable Exp}\ \texttt{)} \\
 & | & \texttt{(begin}\ \textit{Exp}_1 \ldots \textit{Exp}_n\ \textit{Exp}\ \texttt{)}
\end{array}
$$

$$
\begin{array}{rcl}
\textit{Constant} & ::= & \textit{IntegerNumeral} \\
\textit{UnaryOp} & ::= & \texttt{zero?} \mid \texttt{succ} \mid \texttt{pred}
\end{array}
$$

Write in Java a program which recognizes programs in the above subset, and prints a type for that program or decides that none exists. Rewrite the programs from questions 1–6 into Scheme, and as part of your submission, include the results of running your solution on these Scheme programs. Submit also the type rules and the constraints you use for `set!` and for `begin`.