9 Semantics of Programming Languages (nk480)

Many languages (like C and Java) support *coercions*, in which values of one datatype (e.g., machine integers) can be used where values of another datatype (e.g., floating point numbers) are expected, by having the compiler silently insert code to convert from one type to another. Suppose we have a language with the following grammar of types:

$$T ::= \mathsf{int} \mid \mathsf{bool} \mid \mathsf{string} \mid T \times T' \mid T \to T'$$

Suppose we then define a subtyping relation as follows:

$$\frac{}{T \leq T} \qquad \frac{T \leq T' \qquad T' \leq T''}{T \leq T''}$$

$$\frac{T_1 \leq T_1' \qquad T_2 \leq T_2'}{T_1 \times T_2 \leq T_1' \times T_2'} \quad \frac{T_1' \leq T_1 \qquad T_2 \leq T_2'}{T_1 \to T_2 \leq T_1' \to T_2'}$$

$$\frac{}{\mathsf{bool} \leq \mathsf{string}} \qquad \frac{}{\mathsf{int} \leq \mathsf{string}}$$

$$\frac{}{\mathsf{bool} \leq \mathsf{int}}$$

(*a*) Assuming the existence of functions `bool_to_string`, `int_to_string`, and `bool_to_int`, adapt the relation above to define a new relation $T \leq T' \rightsquigarrow e$, where e is a *coercion*, a closed function of type $T \to T'$. (You may use ML or lambda-calculus notation to define the coercions e.) [10 marks]

(*b*) Explain what this relation could be used for in a language implementation. [2 marks]

(*c*) Give definitions of `bool_to_string` and `bool_to_int`, and then use the relation you defined to give two subtyping derivations $\mathsf{bool} \leq \mathsf{string} \rightsquigarrow e_1$ and $\mathsf{bool} \leq \mathsf{string} \rightsquigarrow e_2$ such that e_1 and e_2 have different behaviour. [5 marks]

(*d*) What problem would this lead to in a language implementation? [3 marks]