COMPUTER SCIENCE TRIPOS Part II – 2017 – Paper 8

15 Types (AMP)

(*a*) (*i*) Explain the difference with respect to polymorphic occurrences of variables between the way let-bound and lambda-bound variables are treated in languages featuring ML-style polymorphism. Illustrate your answer by explaining why in the Mini-ML language one of the two expressions

$$\lambda g\,(\texttt{let}\,f = \lambda x\,(\texttt{let}\,u = g\,x\,\texttt{in}\,x)\,\texttt{in}\,f\,f)$$
$$\texttt{let}\,f = \lambda x\,(\texttt{let}\,u = x\,\texttt{in}\,x)\,\texttt{in}\,f\,f$$

is typeable and the other is not. As part of your answer you should define what it means for a type to be a specialisation of a type scheme and give the Mini-ML typing rules for variables, function abstractions, function application and let-expressions. [14 marks]

(*ii*) What property of the type system is lost if one attempts to treat polymorphic occurrences of lambda-bound variables in the same way as let-bound ones? [1 mark]

(*b*) Give the typing rules for expressions associated with ML-style reference types, namely reference creation $\texttt{ref}\,M$, reference look-up $!M$ and assignment $M := M'$ expressions. What is the *value-restriction* that is imposed on the typing rule for let-expressions in the presence of these forms of expression and what is its purpose? [5 marks]