COMPUTER SCIENCE TRIPOS Part II – 2025 – Paper 9

13 Types (nk480)

(*a*) Consider the following OCaml type:

```
type bexp = True | False | Not of bexp | And of bexp * bexp
```

In this question we will look at its encoding in System F.

(*i*) Give a suitable System F type for a Church encoding of the `bexp` type. [2 marks]

(*ii*) Give an implementation of the `True`, `False`, `Not` and `And` constructors for this encoding. [4 marks]

(*iii*) Give the type and encoding of the recursive eliminator named fold for this tree type. [2 marks]

(*iv*) Give reduction rules for fold. [3 marks]

(*v*) For the `And` case, show that your encoding models the reduction rule correctly. [4 marks]

(*b*) (*i*) Using the simply-typed lambda calculus augmented with state and integers, write a function $\mathsf{count} : ((\mathsf{unit} \to \mathsf{unit}) \to \mathsf{unit}) \to (\mathsf{unit} \to \mathsf{unit}) \to \mathsf{int}$, such that $\mathsf{count}\, k\, f$ computes $k\, f$, and returns the number of times k invokes the function f. [3 marks]

(*ii*) In the pure, total simply-typed lambda calculus with integers, characterise the behaviour of a function $h : ((\mathsf{unit} \to \mathsf{unit}) \to \mathsf{unit}) \to (\mathsf{unit} \to \mathsf{unit}) \to \mathsf{int}$ in terms of its arguments k and f. [2 marks]