OLLSCOIL NA hÉIREANN MÁ NUAD

THE NATIONAL UNIVERSITY OF IRELAND MAYNOOTH

JANUARY 2014 EXAMINATION

CS424

Programming Language Design & Language Semantics

Dr. D. Charles, Dr. A. Winstanley, Prof. B. Pearlmutter

Time allowed: 2 hours

Answer all four questions

All questions carry equal marks

[25 marks]

1 What will the following return, when entered into a Scheme interpreter?

Define a higher-order function deep-fetch which takes a predicate and an s-expression, and returns a list of all atoms inside the given s-expression which pass the given predicate.

Examples:

(deep-fetch number? '(the (quick 6 fox 8 9) slick 2))
-> (6 8 9 2)

(deep-fetch symbol? '(the (quick 6 fox 8 9) slick 2))
-> (the quick fox slick)

[25 marks]

2 Define a higher-order function `mapEveryOther` with the type

`mapEveryOther` :: (a->a) -> [a] -> [a]

which applies the given function to *every other* element of the given list, starting with the first element.

Examples:

`mapEveryOther` (+1000) [0,1,2,3,4,5,6] =
1000,1,1002,3,1004,5,1006]

It should work on lists of either even or odd length.

[25 marks]

3 Define a predicate noah which takes elements from a list two-by-two.
E.g.,

noah([a,b,c,d,e,f,g,h], [[a,b],[c,d],[e,f],[g,h]]).

[25 marks]

4 Reduction of a well-typed expression in the simply typed lambda calculus always terminates, while expressions in the untyped lambda calculus can reduce indefinitely without termination.

(a) Give an example of an expression whose reduction fails to terminate in the untyped lambda calculus;

(b) prove that it cannot be well-typed by showing that type inference results in incompatible constraints on the inferred types.