Programming Language Concepts, January–April 2017

λ-calculus

Assignment, 2 April, 2017
Due: *10 April, 2017*

Note: Only electronic submissions accepted, via Moodle.

Notation:

- In the untyped lambda calculus, let $\langle n \rangle$ be the Church numeral encoding of the number n—that is, $\langle n \rangle = \lambda fx.(f^n x)$.
- Also, let
  - $\langle \text{true} \rangle = \lambda xy.x$.
  - $\langle \text{false} \rangle = \lambda xy.y$.
  - $\langle \text{if } b \text{ then } x \text{ else } y \rangle = \lambda bxy.bxy$.

1. Verify that $\lambda pq.(pq)$ encodes the exponentiation function for Church numerals.
2. Find an encoding for the predecessor function given by:
   (a) $\text{pred}(0) = 0$.
   (b) $\text{pred}(n) = n - 1$, for $n > 0$.
3. Write an expression in the second order polymorphic typed λ-calculus (System F) corresponding to the Haskell function.

```
twice f x = f (f x)
```

   Derive its type using the type inference rules for the 2nd order polymorphic typed lambda calculus.
4. Unify the following pairs of formulas, if possible. Explain your answer in terms of the unification algorithm discussed in class.
   (a) $p(x, g(f(a)), f(x))$ and $p(f(y), z, y)$.
   (b) $p(a, x, f(g(y)))$ and $p(z, h(z, u), f(u))$.