Assignment #4

ECEN 5023, CSCI 7135

Due May 6, 2008

1. Implement in OCaml a type checker and interpreter for a language with the following features:
   - Booleans and natural numbers (Figures 8-1 and 8-2).
   - First-class functions (lambdas) (Figure 9-1). The type annotations on function parameters are optional, so if they are left off you should infer their types in your inference algorithm.
   - References (Figure 13-1)
   - Records (Figure 11-7) and variants (Figure 11-11).
2. Add support for two or more of the following options (your choice):
   - Let-polymorphism and type inference (described in Section 22.7 of the text). Do not use term substitution to implement let-polymorphism in the type inference algorithm. Instead use the more efficient approach described on the bottom of page 333.
   - recursive types
   - first-class polymorphism
   - subtyping, subsumption, and bounded polymorphism
3. Write a type checker and interpreter for the simply typed lambda calculus in this language that you've implemented.
4. Bonus: if you've implemented recursive types and bounded polymorphism, then you can implement the object encoding of Abadi, Cardelli, and Viswanathan in their paper "An interpretation of objects and object types".

A few remarks:

- You may use any of the implementations on the web page of the textbook as your starting point.
- Use the exact syntax that is used in the textbook.

- Test cases will be provided on the course web page. You are encouraged to post interesting test cases to the course mailing list.
- You may work in teams of up to two people. You may use any resources that you find on the Internet including papers and code.
- Don't wait till the last minute. You'll need to work steadily on this over the next month to have any hope of finishing.