Department of Computer Science and Engineering
University of California at San Diego

CSE 230
Winter 1998

Problem Set 2

DUE TUESDAY JANUARY 20 IN CLASS.

1. In mathematical logic, types were invented by Bertrand Russell to overcome a paradox of naive set theory that he described in 1902 in a letter to Gottlob Frege. The paradox is

   Let $W = \{X : X \notin X\}$. Is it the case that $W \in W$?

   In naive set theory, all entities are sets, and there is one basic binary predicate, namely membership $\in$. There are a number of ways of creating sets. Notably, a set can be empty, the complement of another set, or the union of two other sets. A set can also be defined intensionally, as containing all sets satisfying some logical formula. This method of definition is used for W above.

   Define an ML datatype to represent the sets of naive set theory, and define an appropriate member function. Create an ML value `w` to represent W, and a value `v` to represent the set of all sets that are members of themselves. Does the ML type system allow you to evaluate `member v v`? And `member w w`? If so, what happens? If not, why not?

2. Explain informally what the class of "primitive recursive" functions is, and what the class of "general recursive" functions is. How is the distinction between these respected in Pascal, but not in C? Is the distinction respected in ML?

   Inspired by your answers to the questions above, show that ML is Turing-complete.

References

An excellent source for both problems above is the series of pages starting at

`http://www.netaxs.com/people/nerp/automata/church0.html`.

Primitive recursion and general recursion are explained in many books, including *Computability and Logic* by George S. Boolos and Richard C. Jeffrey, third edition, Cambridge University Press, 1989, where the basic concepts are at the bottom of pages 73 and 81. Note that general recursive functions are sometimes called just "recursive".