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

CSE 230
Winter 1997

Final Examination

TUESDAY MARCH 18, 11:30AM TO 2:30PM.

Your name:

This exam has 5 pages and 50 points in total. Look through the whole exam and answer the questions you find easiest first. Answer each question in the space following it, using the backs of the pages as extra space if necessary.

You may use your own personal lecture notes, the online class lecture notes, your own old assignments, and all handouts and class textbooks. Other books or notes may not be used.

1. (i) [6 points] Comment on the relative benefits and drawbacks of programming in C, ML, and Java.

   In your answer, discuss relevant historical trends and similarities and differences between languages. Use ideas introduced in CSE230 as appropriate. Keep in mind the contrast between programming by a team versus by one person, and the contrast between exploratory hacking and the engineering of long-lasting, robust efficient software.

   (ii) [3 points] Make some reasonable predictions about the design of future programming languages.

2. [8 points] Using one or more higher-order polymorphic functions, write elegant ML functions of type α `list -> `α `list` for the two tasks below.

   (i) [4 points] Compare adjacent list elements and remove all adjacent duplicates. For example, given `[1,2,1,1,1,3,3]` this function should return `[1,2,1,3]`.

   (ii) [4 points] Compare adjacent list elements and leave just one copy of each duplicated element. For example `[1,2,1,1,1,3,3]` yields `[1,3]`.

3. [8 points] Consider a typical imperative, sequential, non-higher-order programming language with nested procedure declarations and static scoping, such as Pascal without procedure or function parameters.

   (i) [4 points] Explain in English sensible semantics for a `goto` from inside a procedure `P` when the destination is outside `P`. Be concise but consider all possible cases for where the destination may be. Use examples as appropriate.

   (ii) [4 points] Why must such a language be implemented using closures if `goto` destinations can be passed as parameters?

4. [16 points] Let L be a language of expressions with the following features: literal constants `true` and `false`, integer numerals, addition, multiplication, equality, and *if-then-else* operators, variables x, y, z, etc., and non-recursive lambda expressions. Lambda expressions use call-by-value and static scoping. For example the following expression has value 32:

```
(lambda x.
        (lambda y.
                (lambda x.x+x)(y*y)
        )(x+x)
)2
```

(i) [4 points] Give an abstract syntax grammar for L that enforces the obvious rules of type correctness concerning booleans and integers.

(ii) [4 points] Using your abstract syntax, give a denotational semantics $M[\cdot]$ for expressions of the language L that do not contain free variables. Be sure to use call-by-value and static scoping.

(iii) [8 points] A function $f(m, n)$ is *implicitly definable* in a language if there is an expression E in the language with free variables x and y such that for all m and n, if E' is E with x replaced by m and y replaced by n, then $M[E'] = f(m, n)$.

Using structural induction, prove that the integer exponential function m^n for $m \geq 1$ and $n \geq 1$ is *not* implicitly definable in the language L.

5. [9 points] (i) [3 points] Write down the lambda expressions usually named $true$ and $false$, and give an appropriate lambda expression corresponding to not.

   (ii) [3 points] Explain the connection between Russell's paradox in set theory and the lambda expression $R \equiv (\lambda x.not(x\,x))(\lambda x.not(x\,x))$.

   (iii) [3 points] Show that $R = false$ if and only if $R = true$. Why is this consistent with the fact that $true \neq false$? (Here $=$ means the lambda calculus notion of equality.)