University of Toronto
St. George Campus

Computer Science 324
Fall 2003

Midterm Exam
Section L5101
30 October, 2003

DO NOT OPEN THIS BOOKLET UNTIL YOU ARE TOLD TO DO SO

Last Name:

First Name:

Student Number:

Rules:

1. Legibly write your name on this page and every page that you wish to be marked or returned. The pages of this exam will be separated during marking. *Unnamed pages will not be marked.*
2. There are 2 problems of equal weight.
3. Total time is 50 minutes.
4. The exam is closed book, and no aids of any kind are allowed.
5. Write your answers directly on the exam in the space provided, or on the blank pages provided at the end. For each blank page that you use, write the number of the problem that you are solving on that page. Do not write the answer to more than one problem on the same blank page.
6. For rough work, you may use the back of any page. *These will not be marked.*

Do not write below this line.

1:

2:

Total:

Name:

1. (20 marks) In class, we discussed the `reduce` function in Scheme, which combines a list of values into a single value by iteratively applying a binary function. For example:

```
(reduce + ‘(1 2 3 4 5) 0)
```

computes the sum, `(+ 1 (+ 2 (+ 3 (+ 4 (+ 5 0))))) = 15`, and:

```
(reduce append ‘((1 2 3) (4 5) (6 7)) ())
```

computes value of the expression:

```
(append ‘(1 2 3) (append ‘(4 5) (append ‘(6 7) ())))
```

which is `(1 2 3 4 5 6 7)`.

a. (12 marks) Define `reduce` **in ML** such that `reduce:('a * 'b -> 'b) * 'a list * 'b -> 'b`.
b. (8 marks) Define an ML function, `length:'a list -> int` which calculates the length of a list by calling `reduce` with an anonymous function, i.e., a lambda expression, as its first argument.

Name:

2. (20 marks) With reference to the following two ML function definitions:

```
1 fun sum [] = 0
2   | sum (i::T) = i + sum(T);

3 fun double [] = []
4   | double (i::T) = ((2*i)::double(T));
```

prove that `sum(double L)` $= 2 *$ `sum(L)`, for all `L:int list`. You may assume that both of these functions terminate on every well-typed input.

Note: Each line of your proof must come with a justification. The only justifications you need for this proof are:

- `1-4`, one of the above definition line numbers,
- `IH`, the inductive hypothesis,
- `arith` for any rule of arithmetic (in other words, you can assume that `+` and `*` work as we expect), and
- `case` for steps that are justified by the assumptions of the current case you are proving.

Your proof should be inductive, which means that it will use `IH` at least once and it will have at least two cases, a base case and a recursive case.

Name:
Blank page: Problem ____________

Name:
Blank page: Problem ___________