

**Honors Programming Languages  
G22.3110 Spring 2008**

**Final Exam  
In-Class Portion**

Write the answers to these questions in a blue book and turn it in before you leave the classroom.

1. (a) Consider the following Ada program. Even though it has two concurrently executing tasks, the program would make poor use of a multiprocessor computer. State why.

```
procedure foo is
  task produce;
  task add is entry input(x:integer); end add;

  task body produce is
  begin
    for i in 1..100 loop
      add.input(i*i);
    end loop;
    lotsa_work; -- a very long running procedure
  end produce;

  task body add is
    num: integer;
  begin
    for j in 1..100 loop
      accept input(x:integer) do
        num := x;
      end input;
      more_work(num); -- a long running procedure
    end loop;
  end add;

begin
  null;
end foo;
```

- (b) Modify the program so that its performance on a multiprocessor would be improved, if possible. Do not add any additional tasks (hint: use a feature added in Ada95 that did not exist in Ada83).
2. (a) Write, in ML or Scheme, the procedure **reduce** that takes three parameters: a binary function  $f$ , a list  $L$ , and a value  $v$ . If  $L$  is empty, then **reduce** should return  $v$ . Otherwise, assuming  $L$  contains the elements  $x_1, x_2, \dots, x_{n-1}, x_n$ , **reduce** should return

$$f(x_1, f(x_2, \dots, f(x_{n-1}, f(x_n, v)) \dots))$$

For example, in Scheme,

```
(reduce + '(1 2 3) 0)
```

would return 6, namely the result of  $(+ 1 (+ 2 (+ 3 0)))$ , and, in ML,

```
reduce (fn x => (fn y => x andalso y)) [true,false,true] true
```

would return `false`.

- (b) If `reduce` were written in ML, what would its type be?
  - (c) Write a sorting procedure, using `reduce`, whose parameter is an unsorted list of integers and which returns a sorted list. You can write auxiliary procedures if necessary.
3. Write a short program fragment that would execute in linear time using any popular parameter passing mechanism other than pass-by-name but runs in quadratic time using pass-by-name.
4. The first Church-Rosser Theorem states that if terms  $X$  and  $Y$  are inter-convertible, then there exists a term  $Z$  such that  $X$  can be reduced to  $Z$  (using only reduction) and  $Y$  can be reduced to  $Z$ . State and prove the important corollary of the first Church-Rosser theorem that tells us about the result of picking one reduction order or another when reducing a term to a normal form.
5. (a) What features of the idealized logic program interpreter, as given in the class notes, are unimplementable?
- (b) Suppose Prolog implementations always attempted to produce *all* solutions to a query. Which of the features that you mention in part (a) would provide no additional benefit, even if it could be implemented?
- (c) In Prolog, assuming a binary tree structure is defined using terms of the form `leaf(X)` and `node(X,Y)`, write the fringe procedure. For example, the fringe of the tree represented by the term

```
node(leaf(3), node(leaf(4), leaf(5)))
```

is `[3,4,5]`.

- (d) Is the function reversible? That is, can it be used to generate a tree whose fringe is given? Can it be used to generate all trees with that fringe? Justify your answers.