CMPT 340 **Programming Systems and Languages** **Winter 2003**

MidTerm Examination
October 28, 2003

Time: 45 Minutes
Total marks: 30

Question 1 *(5 marks)*

Consider the following Prolog program.

```
rel([A]).
rel([a,b|Cs]) :-
  rel([b|Cs]).
rel([b,a|Cs]) :-
  rel([a|Cs]).
rel([c,b|Cs]) :-
  rel([b|Cs]).
```

Draw the search tree for the query `rel([X,b,Z]).`

Question 2 *(5 marks)*

1. Consider the following Haskell program.

```
data Silly = This Int | That Bool
run :: Silly -> Silly
run (This i)
  | i + 5 > i*i    = That True
  | otherwise      = That False
run (That c)
  | c == True      = This 5
  | otherwise      = run (This 7)
```

   What is the value of the following expressions:

   (a) `run (This 1).`

   (b) `run (That False).`

2. What is the type signature for the following function definitions:

   (a) `f1 f2 f3 a = f2 (f3 a)`

   (b)
```
fun x y [] = []
fun x y (a:as)
  | x == a     = (y:fun x y as)
  | otherwise  = (a:fun x y as)
```

Question 3 *(5 marks)*

Consider the following Prolog program. The intended meaning is documented in the code.

```
% minimum(X,Y,Z)
% X, Y, Z are all numbers
% the number Z is the smaller of X and Y
minimum(X,Y,X) :-
  X < Y.
minimum(X,Y,Y).
```

1. Define what it means for a Prolog program to be correct and complete.
2. If the given program is correct, give an informal proof. Or else, fix the program, and show that your version is correct. Assume that the Prolog builtin `X < Y` is correct.
3. If the given program is complete, give an informal proof. Or else, fix the program, and show that your version is complete. Assume that the Prolog builtin `X < Y` is complete.

Question 4 *(10 marks)*

Answer either **one** of the following two questions using Prolog, and the **other** using Haskell.

1. Write a program `minmax` that returns the minimum and maximum element of a list of numbers. For example, given `[4,7,1,3,2,5]`, `1` is the minimum element, and `7` is the maximum element. Note that for a list of one element, the maximum and the minimum are the same.

   For Prolog, you should write `minmax/3`.
   For Haskell, you should write `minmax :: [Int] -> (Int,Int)`

2. Write a program `partition` that splits a list of numbers into ~~three~~ two lists: the numbers less than, or greater than a given number. For example, given the list `[7,1,3,2,5]` and the number `4`, the list `[1,3,2]` are smaller and `[7,5]` are all larger. Note: the order of the elements in the smaller lists is not important. You may assume that the given number does not appear in the list.

   For Prolog, write `partition/4`.
   For Haskell, write `partition :: [Int] -> Int -> ([Int],[Int])`

Question 5 *(5 marks)*

You may solve this problem using either Prolog or Haskell.

The i-prefix of a list L is a list containing the first i elements of L. For example, `[1,2]` is a 2-prefix of the list `[1,2,3,4,5]`, and `[1,2,3,4,5]` is the 5-prefix of itself. The empty list `[]` is a 0-prefix of every list.

Write a program (in Haskell or Prolog) that, given a list, returns a new list of all i-prefixes, $i = 0, \cdots, l$ where l is the length of the given list. For example, given `[1,2,3,4,5]`, the list of all i-prefixes is `[ [], [1], [1,2], [1,2,3], [1,2,3,4], [1,2,3,4,5] ]`.

You may use the builtin programs `reverse/2` (Prolog) or `reverse :: [a] -> [a]` (Haskell) to reverse a list. You are not required to use it, however.