Midterm Exam, COMP3031, Fall 2014

Date	Oct 21, 2014 (Tuesday)
Time	16:30-17:50
Instructions:	(a) This exam contains five problems, counting for a total of 100 points.
	(b) Write ALL answers in the exam book. Do not use any other papers.

	Problem	Points
Name:		
Student ID:	1.	
ITSC Account:	2.	
	3.	
	4.	
	5.	

Total:

Problem 1 (10 pts) What is the value of each of the following SML expressions (a)-(b)?

```
(*a*)
fun calc2 x y = let val x=y-1 val y=x+2 in x*y end;

(*b*)
datatype tree = Empty | Node of string * tree * tree;

fun pr(x,Empty) = x
  | pr(x,Node(y,left,right) ) =
     if (size x) < (size y) then pr(x,left) ^ pr(y,right)
     else pr(y,left) ^ pr(x,right);
```

(a)

```
calc2 1 8;
```

(b)

```
pr("12", Node("3", Node("45",Empty, Empty),
Node("6", Node("7",Empty,Empty), Node("890",Empty,Empty))));
```

Problem 2 (15 pts) What is the type of each of the following SML functions (a)-(c)?

(a)

```
fun e [] n = "false"
  | e (h::t) n = if h = n then "true"
                            else "false" ^ (e t n);
```

(b)

```
fun r [] s = s 0.0 |
    r (h::t) s = s (h (r t s));
```

(c)

```
fun sp c L R p [] = (L, R)|
    sp c L R p (h::t) =
       if c(h,p)
       then sp c (h::L) R p t
       else sp c L (h::R) p t;
```

Problem 3 (30 pts) Write the following SML functions (a)-(b).

(a) `listDiff = fn : ''a list -> ''a list -> ''a list`. Given a list of equality-comparable elements, `listDiff` returns a list of elements that are in the first list but not in the second list. Examples:

```
- listDiff ["a", "b", "c"] ["b"];
val it = ["a","c"] : string list
- listDiff nil [1,2,3];
val it = [] : int list
- listDiff [1,2,3] [3,4,5,6];
val it = [1,2] : int list
- listDiff [1,2,3] nil;
val it = [1,2,3] : int list
- listDiff [1,3,3] [2,3];
val it = [1] : int list
- listDiff [3,3] [3];
val it = [] : int list
```

(b) `sublistReverse = fn : 'a list list -> 'a list list`. Given a list of lists, `sublistReverse` returns a list such that (1) the order of all sublists is reversed; and (2) the order of elements in each sublist is reversed. Examples:

```
- sublistReverse [nil, ["a"], ["b","c"]];
val it = [["c","b"],["a"],[]] : string list list
- sublistReverse [[1,2,3], [4,5],[6]];
val it = [[6],[5,4],[3,2,1]] : int list list
- sublistReverse [[6]];
val it = [[6]] : int list list
- sublistReverse nil;
val it = [] : ?.X1 list list
```

Problem 4 (15 pts) Consider the following grammar in BNF with `<S>` being the starting non-terminal:

```
<S>::= <I>.<I>.<I>.<I>
<I>::= <D> | <N><D> |1<D><D> | 2<F><D> |25<V>
<D>::= 0|1|2|3|4|5|6|7|8|9
<N>::= 1|2|3|4|5|6|7|8|9
<F>::= 0|1|2|3|4
<V>::= 0|1|2|3|4|5
```

(a) Determine whether the string "143.89.40.4" belongs to the language generated by the grammar. If your answer is yes, write a derivation of the string based on the grammar; If your answer is no, just say so and no explanation is needed.

(b) Is this grammar ambiguous? If your answer is yes, write an **unambiguous** grammar in BNF to represent the language; if your answer is no, just say so and no explanation is needed.

Problem 5 (30 pts) Consider the following definition of list expressions:

- Lists "A", "B", "C", "D", and "E" are list expressions.
- Given list expression A, "~"A, whose elements are A's in reversed order, is a list expression.
- Given list experssion A, "("A")" is a list expression.
- Given list expression A,"FILTER" A and "MAP" A are list expressions.
- Given list expressions A and B, A "MERGE" B is a list expression.
- Given list expressions A and B, A "DIFF" B is a list expression.

The operators on list expressions obey the following rules in **decreasing precedence** (operators on the same line have the same level of precedence):

```
()
~ FILTER MAP       (right associative)
MERGE DIFF        (left associative)
```

(a) Write an **unambiguous** context-free grammar in BNF for such list expressions, preserving the precedence and associativity of the list operators.

(b) Draw the **tree representation** of the following list expression:
" `~MAP A DIFF FILTER B MERGE (C MERGE FILTER MAP (D DIFF E))`".

Blank Page