COMP 3031 Assignment 3
Logic Programming
Fall 2014

Due: 5:00pm on Nov. 27, 2014 (Thursday)

Instructions

- There are *five* problems in this assignment. Each problem counts for two points.
- Write your prolog program according to the definition of the problem, with the same predicate name and number of arguments as specified. Write all the solutions in a single file named "ass3.pl".
- Write your *name, ITSC account name, and student ID* (using /* comment */) on the first line of your submission file.
- Submit *ass3.pl* through the course assignment submission system (CASS) of COMP3031 (do not submit to other courses) before the deadline: `https://course.cse.ust.hk/cass/submit.html`.
- Instructions on using CASS are available at: `http://cssystem.cse.ust.hk/home.php?docbase=UGuides/cass&req_url=UGuides/cass/student.html`.
- *No* late submissions will be accepted.
- Your submission will be run on a lab 2 machine with the following command:

```
?- [ass3].
```

- *Make sure your submission can be run using the above command. If it can not be executed, you may get 0 marks for this assignment!*

For all the following problems, the inputs are given as specified.

Question 1. List with a mirror

Define a predicate $listmirror(L1, L2)$, where $L2$ is the concatenation of $L1$ and its mirror. At least one of $L1$ and $L2$ is given.

Examples:

```
?- listmirror([1,2,3],L).
L = [1, 2, 3, 3, 2, 1].

?- listmirror([],[1]).
false.

?- listmirror(L, [aa,bb,cc,cc,bb,aa]).
L = [aa, bb, cc].
```

Question 2. List diff

Define a predicate $listdiff(L1, L2, L3)$, where $L3$ is a list of elements that appear in $L1$ but not in $L2$. Both $L1$ and $L2$ are given, $L3$ can be a given list or a variable.

Examples:

```
?- listdiff([],[1,2],L).
L = [].

?- listdiff([1,2,3,d,t],[1,d],L).
L = [2, 3, t].

?- listdiff([1,y,2],[2],[1,y]).
true.
```

Question 3. Insert a number to a sorted list

Define a predicate $insert(X, L1, L2)$, in which X is a given number, $L1$ is a given sorted list of numbers, and $L2$ is the sorted list from inserting X into $L1$ at the correct position. $L2$ can be a given list of numbers or a variable.

Examples:

```
?- insert(3,[1,4,7],L).
L = [1, 3, 4, 7].

?- insert(3,[1,4,7],[1,3,4,7]).
true.

?- insert(7, [], L).
L = [7].
```

Question 4. Binary string generator

Define a predicate $bitsgen(X, L)$, which generates all the possible binary strings with X bits. X is a given non-negative integer and L is a given list or a variable.

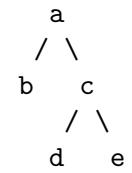
Examples:

```
?- bitsgen(2, L).
L = [0, 0] ;
L = [0, 1] ;
L = [1, 0] ;
L = [1, 1] ;
false.

?- bitsgen(3, L).
L = [0, 0, 0] ;
L = [0, 0, 1] ;
L = [0, 1, 0] ;
L = [0, 1, 1] ;
L = [1, 0, 0] ;
L = [1, 0, 1] ;
L = [1, 1, 0] ;
L = [1, 1, 1] ;
false.
```

Question 5. Traverse a binary tree

We represent a binary tree in Prolog in predicate $t(X, L, R)$, where X is the root node, L and R denote the left and right subtree, respectively. We use atom "nil" to represent an empty tree node. For example, a tree $T = t(a, t(b, nil, nil), t(c, t(d, nil, nil), t(e, nil, nil)))$ looks like:



Define a predicate $preorder(T, L)$, which constructs the preorder sequence of binary tree T. T is a given binary tree and L is either a given list or a variable.

```
?- preorder(t(a, t(b, nil, nil), t(c, t(d, nil, nil),
t(e, nil, nil))), L).
L = [a, b, c, d, e].

?- preorder(t(a, nil, nil), L).
L = [a].

?- preorder(t(a, nil, nil), [a]).
true.

?- preorder(t(a, nil, nil), [a,b]).
false.
```