2013 February 18

Prolog Operator
Example Exam Questions

1.

Consider the following three operators with the properties as stated.

`><`	infix right associative operator, low precedence
`<>`	infix left associative operator, medium precedence
`#`	postfix such that "`operand # #`" is legal, high precedence

A Give a Prolog definition for the three operators. Do not worry about whether the precedence values you give them make sense when compared to other operators in Prolog, they only have to make sense relative to one another.

B For each of the following expressions do only one of the following.

- If the expression is legal, then fully parenthesize the expression to show the order in which the operators are evaluated.
- If the expression is illegal, then explain what is the problem.

```
a   ><   b   #   <>   c   <>   d

a   ><   b   ><   c   #   #   ><   d
```

2.

In Prolog the predicate **op** is used to define unary and binary operators. Define and explain what the parameters to **`op`** are. What general values we can give them and why we choose particular values.

3.

In order to do simple propositional logic we want to define 3 operators:

- & (and)
- v (or)
- ~ (not)

We want & to be left associative, v to be right associative and ~ ~ x to be a legal statement. The ~ operator should have the highest precedence, followed by &, followed by v. Do not worry about whether the precedence values make sense when compared to other operators in Prolog they only have to make sense relative to one another.

You are given a database of propositional variables which have values given by the value predicate. The database is as follows:

```
value( a, 1 ) :- !.
value( c, 1 ) :- !.
value( d, 1 ) :- !.
value( f, 1 ) :- !.
value( g, 1 ) :- !.
value( V, 0 ) :- atom(V) , \+ member(V, [a, c, d, f, g]).
```

Define the 3 operators described above and then define predicates to give them meaning. The following is an example.

```
?- ~b.
yes

?- a v b.
yes

?- a & b.
no
```

```
?- a & c.
yes
```

Now that the basic operators work, how do you get the following examples to work?

```
?- a & ~b.
yes

?- a & a & a.
yes
```

4.

Prolog provides a means to define operators using `op(X,Y,Z)`. For example, operators 'has' and 'isa' can be defined so the following syntax could be used to enter rules.

```
Animal has hair :- Animal isa mammal.
```

Give an example definition for 'has' and 'isa'