Prolog Notes

* PROLOG stands for **PRO**gramming **LOG**ic. Prolog is the major example of 4th generation programming language which supports declarative programming paradigm.
* In Declarative language the programmer specifies a goal to be achieved (what the problem is) whereas the Prolog system works out how to achieve it.
* Some important features of Prolog:
  + Declarative (not procedural)
  + Recursion (no “for” or “while” loops)
  + Relations (no functions)
  + Unification
* This type of programing comprises of Facts, Rules and Queries.
* In prolog program we declare facts describing explicit relationships between objects and properties of objects might have.
  + Sneha like ice-cream.
  + Hair is Black.
  + Samsung is Company.
* Rules are implicit relationships between objects such as :
  + Anwesha is daughter of Amlan.
  + Apurba is father of Amlam.
* One can use the system to generate queries by asking questions about relationships between objects, and/or about object properties. Examples:
  + Does Sneha like ice-cream?
  + Apurba is father of whom?
* **Facts are properties of objects or relationships between objects. Example:** 
  + **“Sanjay has phone number 9831126620”.**

**In prolog we write it in this way:**

**phoneno(Sanjay,9831126620).**

* Some other important factors to consider while prolog programming :
  + Names of properties/relationships begin with lower-case letters.
  + The relationships name appears as the first term.
  + Objects appear as comma-separated arguments within parentheses.
  + A period “.” Must ends a fact.
  + Objects also begin with lower-case letters. They also can begin with digits like 1234, and can be strings of characters enclosed in quotes such as color(drawing\_pencil,’blue’).
  + phoneno(Sanjay,9831126620) is also called a predicate or clause.
* Prolog is good for prototyping but not as efficient as C.

**RULES:**

* Consider the following case which producws a general rule:
  + One teacher will guide a student if that student studies that very course\_id on which the teacher teaches.
  + **In Prolog this will be written as :**

**supervisor(Teacher, Student) :- teaches(Teacher, course\_id), studies(student,course\_id).**

* Facts are unit clauses and rules are non-unit clauses.
* Variable names will start a capital letter.
* Note:
  + :- means if and only if
  + , comma mean and.

**QUERY OR GOAL**

* Query will be based on facts and values. We can ask questions based on the stored information.
* We want to ask question whether Shubham lectures in course001 or not then we can ask :

?- teaches(shubham, course001).

Yes

* The queries are terminated by full stop(.).
* To answer this query , Prolog consults its database to see if this is a known fact or not.
* We can ask question as follows :

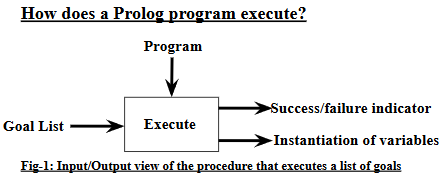
?-teaches(shubham, X).

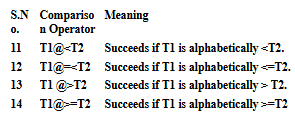
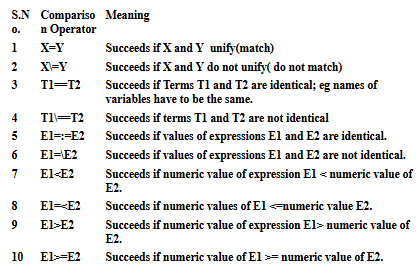
X=course001

* If answer is true/yes , then the query is succeeded.
* If answer is false/no, then the query failed.

**SYNTAX OF A CLAUSE:**

* “:-” means “if” or “is implied by”. It is also called as neck symbol.
* The left hand side of neck is called the head while the right hand side of neck is called body.
* The comma “,” stands for and/conjunction.
* The semicolon “;” stands for or/disjunction.
* A program consists of clauses. These are of 3 types : facts, rules and questions.
* A procedure is a set of clauses about the same relation

.

* **‘is’ evaluation:**

The ‘is’ built in predicate is used in Prolog to force the evaluation of arithmetic expressions. If we write X=2+4, the result is 2+4 and the value will not be evaluated.

Example:

?- X=2+4. (ii) ?- X is 2+4.

X=2+4 X=6

**It is only and always the second argument that is evaluated. This can lead to some strange looking bits of codes by mathematical standards. Example: In Prolog to test whether a number is even, we write 0 is N mod 2, rather than usual mathematical ordering: N mod 2=0.**

**Note: Use ‘is’ where we need to evaluate are arithmetic expression.**

* **not Operator:**

To provide the negation, ‘not’ prefix operator can be placed before any goal. If the original goal fails, the negation goal succeeds.

Example-1:

dog(boxer).

?- not dog(boxer).

No

?- dog(bombay).

No

?not dog(bombay).

Yes

? A is 5, A is 5.

No

* **Disjunction Operator:**

The ‘;/2’ is the disjunction operator. It is an infix operator, which represents ‘or’. It takes two arguments, and both the arguments are goals.

Example:

?- 9<4 ; 10 is 7+3.

true.

?- 5\*5 =:= 25; 12=54+4.

true.

**Problem-1:** marks1.pl: Write a program in Prolog to input Roll, p1,p2,p3. Calculate total= sum of best 2 papers. Average=total/2. calculate Grade using standard procedure.