**Rules**

Rules are the conditional statements about objects and their relationships takes the form **::**

left\_hand\_side :- right\_hand\_side .

This sentence is interpreted as: left\_hand\_side  **if**  right\_hand\_side.

**Example 1:**

If we want to say that john and jane are friends if john likes jane and jane likes john. Then in prolog this friends rule can be written as::

friends(john,jane) :- likes(john,jane), likes(jane,john).

**Example 2:**

             likes(john, X) :- car(X).        // Read as : john likes X if X is a car.   
            friends(X, Y) :- likes(X, Y), likes(Y, X).    // Read as : X and Y are friends if X likes Y and Y likes X.  OR  Two people are friends if they like each other.

**Example 3:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Files Edit Run Compile Options Setup** | | | |
| **Editor**  **Line1 Col1 Indent Insert Work.pro**   Predicates  study (symbol, symbol).  lecture (symbol, symbol).  teach(symbol, symbol).  Clauses  teach(X, Y) :- lecture(X, Z),  study(Y, Z).  lecture(ali, ss).  study(fatan, ss). | | | **----Dialog-----**  Goal: teach(ali, fatan).  Goal: teach(Who, fatan).  Goal: teach(ali, sara). |
| **--------Message---------** | **-------Trace----------** | | |
|  |  |  | |

**Example 4::**

|  |  |  |  |
| --- | --- | --- | --- |
| **Files Edit Run Compile Options Setup** | | | |
| **Editor**  **Line1 Col1 Indent Insert Work.pro**   Predicates  bird(symbol).  animal(symbol).  has\_feathers(symbol).  Clauses  bird(X) :- animal(X),has\_feathers(X).  animal(lion).  animal(sparrow).  has\_feathers(sparrow). | | **----Dialog-----**  Goal: bird(lion) .  Goal: bird(X) . | |
| **--------Message---------** | **-------Trace----------** | | |
|  |  | |  |

**Cut**

The cut in Prolog written as ! , it is used to prevent extra solutions being found by Prolog.

no\_parent(adam,0).

no\_parent(eue,0).

no\_parent(X,2).

Goal: no\_parent(adam,X).

X=0

X=2

**Example** 1:: Rebuilt the above example after put Cut ::

|  |  |  |  |
| --- | --- | --- | --- |
| **Files Edit Run Compile Options Setup** | | | |
| **Editor**  **Line1 Col1 Indent Insert Work.pro**  Predicates  no\_parent (symbol, integer).  Clauses  no\_parent(adam,0):- !.  no\_parent(eue,0):- !.  no\_parent(X,2). | | **----Dialog-----**  Goal: no\_parent(adam,0).  yes  Goal: no\_parent(adam,X).  X=0 | |
| **--------Message---------** | **-------Trace----------** | | |
|  |  | |  |

**Example** 2:: Write a program to find the max from two numbers::

|  |  |  |  |
| --- | --- | --- | --- |
| **Files Edit Run Compile Options Setup** | | | |
| **Editor**  **Line1 Col1 Indent Insert Work.pro**  Predicates  max (integer, integer, integer).  Clauses  max(X,Y,Z):- X>Y, Z=X.  max(X,Y,Z):- X<=Y,Z=Y. | | **----Dialog-----**  Goal: max(9,2,B).  B=9  Goal: | |
| **--------Message---------** | **-------Trace----------** | | |
|  |  | |  |

**Example** 3:: Suppose the following program which is show **true** when input **1** , **false** when input **0** else show **error**. correct program if it need.

|  |  |  |  |
| --- | --- | --- | --- |
| **Files Edit Run Compile Options Setup** | | | |
| **Editor**  **Line1 Col1 Indent Insert Work.pro**  Predicates  test (integer, symbol).  Clauses  test(1,true).  test(0,false).  test(X,error). | | **----Dialog-----**  Goal:  Goal: | |
| **--------Message---------** | **-------Trace----------** | | |
|  |  | |  |

Q:: Try to solve the following question ::

**1-** Write a program that inputs a degree if it is greater than 50 show word "pass" else if less show "not pass".

**2-** Write a program that inputs two numbers ( X and Y), if X greater than Y show the result of multiplication of two numbers else if less show result the summation of two numbers.

**3-** Write a program that inputs a number if it is even show word "even" else if odd show "odd".