Q-1. What is Prolog? Explain features, limitation and application of it.

Answer

It supports formal symbolic reasoning that can understand human language. It's developed in 1972 by P.Roussell. It's object-oriented language which will use data about object and their relationship. It's collection of data on facts and the relationship among this fact.

Features:

(1). We can compile standalone program that will be executed on a machine that is not running in Turbo Prolog and it can be distributed to the user.
(2). It's a full complement of standard predicates for many functions such as string operation, random file access, cursor control, graphics and sound.
(3). It's a functional interface which will allow procedural support to be added in prolog system.
(4). We have to declare the variable which one used to provide more secure development control.
(5). Integer, floating point and real arithmetic values are provided.
(6). An integrated editor is provided making the program development, complication and debugging.

Limitations:

(1). It will reflect the structural aspects of procedural language. The variable declaration requirements and the division of programming in 2 sections which will impose the some of the limitation in symbolic processing.
(2). The current version of turbo prolog doesn't support virtual memory. In this memory the size of the program is limited only by the disk space. So, it's limited by the amount of memory so, we can use random file access to overcome the limitation.
(3). All the prolog programs are inefficient for numerical processing.

Application: It's useful for almost any application that requires the form of reasoning. This includes expert system, NLP, Robotics, Game playing simulation.

(1). **Expert System**: This is the program that use inference technique which involves formal reasoning perform by a human expert to solve a problem in a specific area of knowledge. It can diagnosis, analyze and categorize previously define knowledge. So, it's a collection of rules and facts.

(2). **NLP(Natural Language Processing)**: It's a part of expert system which permits non-tech user to describe the problem and resolve them. It helps the computer to gain the knowledge about a human language which are expressed by rules and facts.

(3). **Robotics**: It's a branch of Computer Science which enables the computer to see and manipulate the objects in their environment. It's primarily involved in study and developing sensor system, manipulators and control the object and space oriented problems.

(4). **Game playing and simulation**: Prolog is ideal for game playing and simulation because of formal reasoning. It implies a set of logical rule that will control the action of many classical game like Tower of Hanoi, 8-puzzle, etc.

Program-1. Write a program to establish relation between 2 symbolic variables

```
Files   Edit   Run   Compile   Options   Setup
Editor
Line 1   Col 1   D:\PRG1.PRO   Indent   Insert
domains
        p1,p2=symbol
predicates
        likes(p1,p2)
clauses
        likes(s,y).

Dialog
Yes
Goal: likes(p1,p2)
No
Goal: likes(s,y)
Yes
Goal: likes(y,s)
No
Goal: likes(a,b)
No
Goal:

Message
Compiling D:\PRG1.PRO
Save D:\PRG1.PRO
Compiling D:\PRG1.PRO
likes

Trace

F1-Help F2-Save F3-Load F5-Zoom F6-Next F7-Xcopy F8-Xedit F9-Compile F10-Menu
```

Program-2. Write a program for medical symptoms example

```
Files   Edit   Run   Compile   Options   Setup
Editor
Line 1   Col 1   D:\PRG2.PRO   Indent   Insert
%Prg-2 Write a prolog program for medical symptoms%
domains
        disease,indication=symbol
predicates
        symptoms(disease,indication)
clauses
        symptoms(flu,runnynose).
        symptoms(cold,runnynose).
        symptoms(cold,wateryeyes).
        symptoms(pox,fever).

Dialog
Yes
Goal: symptoms(pox,feve
r)
Yes
Goal: symptoms(cold,fev
er)
No
Goal:

Message
Save D:\PRG1.PRO
Load D:\PRG2.PRO
Compiling D:\PRG2.PRO
symptoms

Trace

F1-Help F2-Save F3-Load F5-Zoom F6-Next F7-Xcopy F8-Xedit F9-Compile F10-Menu
```

Q-2. Define following terms:(a)predicates (b)clauses (c)Goal (d)fact (e)rule (f)domains

Answer

(a)predicates : A predicates will denote a property or a relationship between the objects
Ex. likes(person1,person2) ← **predicate**

(b)clauses: It's a complete subset of first order predicate logic. A clause has a head which is known as facts between name and arguments. It has a body which is known as rules. If the head is true then only the body is executed.
Ex. likes(Aditya,Naina). ← **fact ('.' shows the ending of fact)**

(c)Goal: It will try to satisfy by finding the values of the variables that makes the goal successful. The values are said to be bound to the variables. If prolog is unable to do this then goal fails.
Ex. Goal: likes(Aditya,Naina)
Yes

(d)fact: The fact must start with a predicate and ends with a full-stop('.'). The predicates may be followed by one or more arguments which are enclosed by parenthesis. The arguments can be numbers, variables or list and it's separated by comma(',').
Ex. symptoms(Flu,runnynose).

(e)rule: Rule can be viewed as an extension of fact with added condition that have to be satisfied for it to be true. It consist of 2 part:
The first part is the fact, predicate with argument and the second part is other clause(fact or rules separated by comma(',')) both part is separated by colon('**:-**').
Ex. hypothesis(Charlie,cold):-
symptom(Charlie,fever).

(f)domains: In this we have to declare the objects which can be integer, real, string, char, symbol.
Ex. domains
x,y=integer

Q-3. What is variable in Prolog?

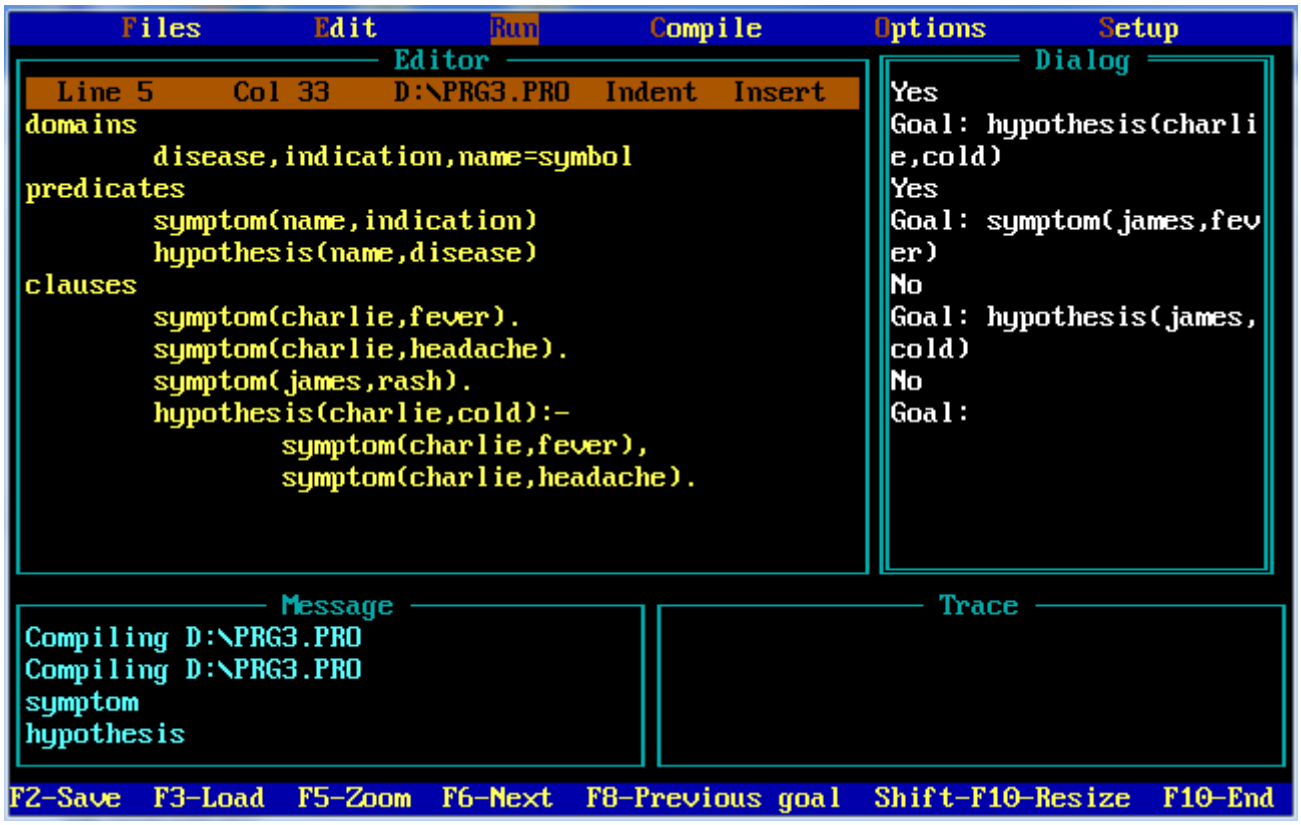
Answer

Variables are normally declared in the domains part.
There are 2 types of variable in Prolog:
1)**Bound variable** : If a variable has a value at a partition time it's said ti be bound.
2)**Free variable** : If a variable doesn't have a value at a particular time it's said to be free.

Program-3. Demonstrate the program using the concept of rule.

Or

Write a prolog program to find the symptoms of patient data disease using the rules.



Q-4. Explain input and output predicate in prolog.

Answer

Input predicates- 4 types

1. **readint**: It can read the **integer** value
2. **readreal**: It can read the **float** value
3. **readchar**: It can read the **char** value
4. **readln**: It can read **string** value

Output predicates- 3 types

1. **write**: The output may be any number, char, string and more
2. **writef**: It's a formatted output function. ***Example at last page***
3. **writedevice**: It will give the output privately to the printer

Program-4. Demonstrate the use of readint and write predicate

Or

Write a prolog program to find out the age of patient.

- **CODE :**

```
Files    Edit    Run    Compile    Options    Setup
 Line 11    Col 49    D:\PRG4.PRO   Indent   Insert
domains
        patient=String
        Age=Integer
predicates
        chkage(patient)
clauses
        chkage(patient):-
        write("What's the age of patient : "),
        readint(Age),
        Age>12,
        write("Patient age is above 12years"),nl.
```

- **OUTPUT:**

```
Files    Edit    Run    Compile    Options    Setup
Yes
Goal: chkage(Soham)
What's the age of patient : 21
Patient age is above 12years
Soham=patient
1 Solution
Goal: chkage(Soham)
What's the age of patient : 10
No Solution
Goal:
```

Program-5. Demonstrate a program for the use of readreal and write predicate.

Or

Write a program to find out the price of an item.

⇨ **CODE:**

```
Files    Edit    Run    Compile    Options    Setup
 Line 9    Col 9    D:\PRG5.PRO   Indent   Insert
domains
        Item=string
        Price=real
predicates
        askprice(Item,Price)
clauses
        askprice(Item,Price):-
        write("Enter the price of ",Item),
        readreal(Price),
        write(Item,"'s ").
```

⇨ **OUTPUT:**

```
Files    Edit    Run    Compile    Options    Setup
Yes
Goal: askprice("Phone",Price)
Enter the price of Phone150.23
Phone's Price=150.23
1 Solution
Goal: askprice("Phone",Price)
Enter the price of Phone58.26
Phone's Price=58.26
1 Solution
Goal: askprice("Laptop",Price)
Enter the price of Laptop6300.58
Laptop's Price=6300.58
1 Solution
Goal:
```

Program-6. Demonstrate a program for the use of readln and write predicate.

Or

Write a prolog program to find whether a city belongs to a state or not?

⇨ **CODE:**

```
Files    Edit    Run    Compile    Options    Setup
 Line 10    Col 34    F:\AILAB\PRG6.PRO   Indent   Insert
        city,state,Reply=string
predicates
        address(city,state)
clauses
        address("Rajkot","Gujarat"):-
                write("Does the Rajkot belongs to Gujarat (Yes/No)? : "),
                readln(Reply),
                Reply="No",
                write("True"),nl.
```

⇨ **OUTPUT:**

```
Files    Edit    Run    Compile    Options    Setup
Yes
Goal: address("Rajkot","Gujarat")
Does the Rajkot belongs to Gujarat (Yes/No)? : No
True
Yes
Goal: address("Rajkot","Gujarat")
Does the Rajkot belongs to Gujarat (Yes/No)? : Yes
No
Goal:
```

Program-7. Demonstrate a program for the use of readchar and write predicate.

Or

Write a prolog program to find whether a city belongs to a state or not?

⇨ **CODE:**

```
Files    Edit    Run    Compile    Options    Setup
Line 11    Col 34    F:\AILAB\PRG7.PRO    Indent    Insert
domains
        City,State=string
        Reply=char
predicates
        address(City,State)
clauses
        address("Rajkot","Gujarat"):-
                write("Is Rajkot capital of Gujarat (Y/N)? : "),
                readchar(Reply),
                Reply='N',
                write("True"),nl._
```

⇨ **OUTPUT:**

```
Files    Edit    Run    Compile    Options    Setup
Yes
Goal: address("Rajkot","Gujarat")
Is Rajkot capital of Gujarat (Y/N)? : True
Yes
Goal: address("Rajkot","Gujarat")
Is Rajkot capital of Gujarat (Y/N)? : No
Goal:
```

Program-8. Demonstrate a program for the use of go predicate.

⇨ **CODE:**

```
Files        Edit        Run        Compile        Options        Setup
 Line 7     Col 21     F:\AILAB\PRG8.PRO   Indent   Insert
domains
        Patient=string
predicates
        go
clauses
        go:-
        clearwindow,_
        write("Enter the name of patient : "),
        readln(Patient),
        write("Patient's name is ",Patient),nl.
```

⇨ **OUTPUT:**

```
Files        Edit        Run        Compile        Options        Setup
Yes
Goal: go
Enter the name of patient : Alex
Patient's name is Alex
Yes
Goal: _
```

Program-9. Demonstrate a program for the use of go and other predicate.

⇨ **CODE:**

```
Files        Edit        Run        Compile        Option
 Line 1     Col 1     F:\AILAB\PRG10.PRO   Indent   Insert
domains
        disease,indication=symbol
        Reply=string
        Patient=string
predicates
        go
        symptom(disease,indication)
clauses
        go:-
        write("Enter the name of patient : "),
        readln(Patient),
        write("Patient's name is ",Patient),nl.
        symptom(cold,wateryeyes):-
        write("Does the patient has cold (Yes/No) ? : "),
        readln(Reply),
        Reply="Yes",
        write("True"),nl.
```

OUTPUT:

```
Files        Edit        Run        Compil
Yes
Goal: go
Enter the name of patient : Alex
Patient's name is Alex
Yes
Goal: symptom(cold,wateryeyes)
Does the patient has cold (Yes/No) ? : Yes
True
Yes
Goal: symptom(cold,wateryeyes)
Does the patient has cold (Yes/No) ? : No
No
Goal:
```

- **writef predicate:** If we want formatted output then we will use writef() predicate.

  Syntax: writef("format",argument,argument(1),....,argument(n))

  %-m.p (ex. %2.7f)

  '-' indicates left justification; right justification is the default.

  'm' specifies the minimum field width

  'p' field determines the precision of a floating-point image (or the maximum number of characters to be printed from a string)

  f - Reals in fixed decimal notation (default)
  e - Reals in exponential notation
  g - Use the shortest format.
  Example at last page

- **writedevice predicate :** Reassigns the current writedevice to the file opened with the given SymbolicFileName, which may be one of the predefined symbolic files (screen and printer) or any userdefined symbolic filename for a file opened for writing or modifying.

  writedevice(printer or screen)

- **inkey predicate :** This predicate will read a single character from the input

  Syntax : inkey(char)

- **keypressed predicate :** It's the predicate which will determine whether a key has been pressed without a character being returned.

  Syntax : keypressed

Define the following terms : (a)backtracking (b)unification

(a) Backtracking :

If any condition fails prolog backtracks to the previous condition and will try to prove it again with another variable then it moves forward again to see if the fail condition will succeed with the newly one prolog moves forward and backward direction through the condition and will try every attempt to get the goal to succeed as many as possible.

(b) Unification :

A term is said to be unify with another term.

1). Both the term appears in predicates that have same no. of argument and both term appear in the same position in their predicate.

2). Both the terms appears as argument of the same type a symbol type can only unify with the symbol type only.

3). All sub terms unify with each other

These are the basic rules of unification :

(i) A variable that is free will unify with any term that satisfies the preceding condition

(ii) A free variable will unify with other free variable. After unifying the 2 variable will act as 1.

(iii) Predicates unify with each other if they have same relation same no of arguments.

Q-5. Explain Cut and Fail predicate with example.

Or

Explain controlling execution in prolog.

Answer

Cut predicate : It removes all the alternatives and then forbid the values otherwise it could be written by method of binding.

It's also very important for recursive process. The symbol of cut predicate is **"!"**. It always succeeds a statement. It will block the backtracking based on a specific condition.

{There are two main uses of the cut:

• When you know in advance that certain possibilities will never give rise to meaningful solutions, so it is a waste of time and storage space to backtrack over them. By using a cut in this situation, the resulting program will run quicker and use less memory.

• When the logic of a program demands the cut. } Extra points

Ex. //Code snippet

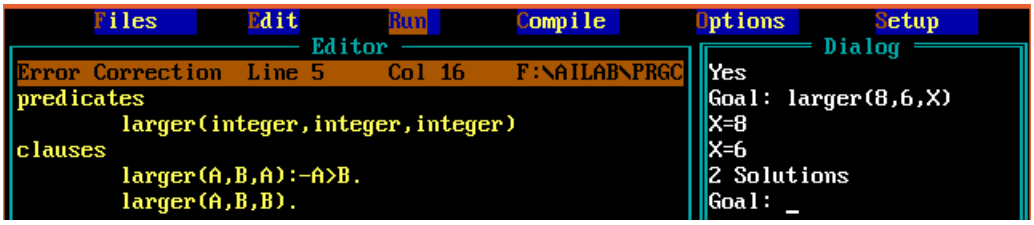
```
a(X):-b(X),!,c(X).
a(X):-d(X).
b(1).
b(4).
c(1).
c(3).
d(4).
```

Explanation: After the cut predicate the whole statement will be going to be true.

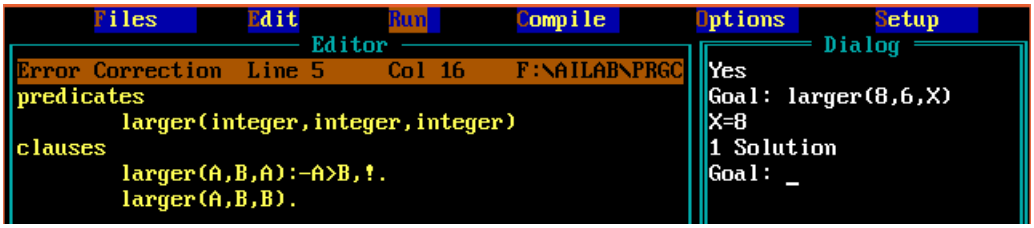
First of all b(x). After that cut predicate indicates the value of b(x) can't be changed that means a(x)=1 and the whole statement is going to be true. So, the value of a(x)=1

Extra example given below for better understanding

Program Without cut predicate:



Program with cut predicate: **(max out of two number)**



Fail predicate: It means the whole statement is going to be false. It will force the backtracking in an attempt to unify with another clause.

It's very useful in recursion and other process.

Ex. //Code snippet

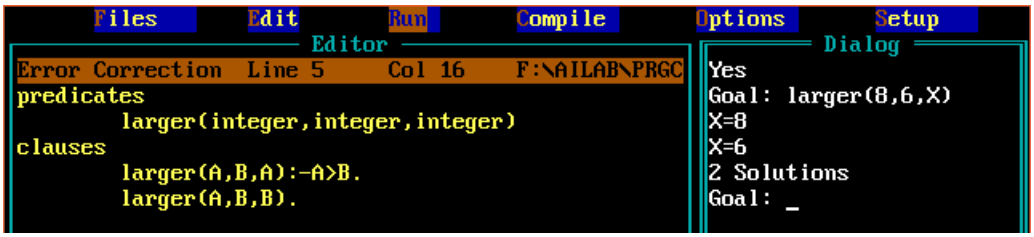
```
a(X):-b(X),c(X),fail.
a(X):-d(X).
b(1).
b(4).
c(1).
c(3).
d(4).
```

Explanation: As a(x)=b(x),c(x),fail. So, the fail statement makes b(x) and c(x) to be failed. So, the value of a(x) can't be b(1),b(4),c(1),c(3).

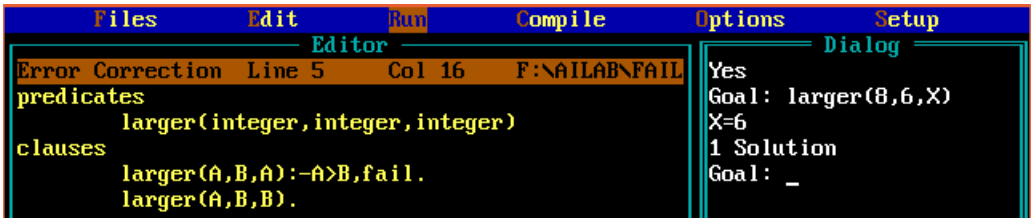
Statement 2 says a(x)=d(x). So, the value of a(x) will be d(4).

Extra example given below for better understanding

Program Without fail predicate:



Program With fail predicate:



Programs for cut and fail predicate.

In Exam if ask then you write any one with output

Program1:

```
Files   Edit   Run   Compile   Options   Setup
Editor
Line 7   Col 23   F:\AILAB\CF1.PRO   Indent   Ins
predicates
a(integer)
b(integer)
c(integer)
d(integer)
clauses
a(X):-b(X),!,c(X),fail.
a(X):-d(X).
b(1).
b(4).
c(1).
c(3).
d(4).
```

```
Dialog
Yes
Goal: a(X)
No Solution
Goal:
```

Program2:

```
Files   Edit   Run   Compile   Options   Setup
Editor
Line 7   Col 16   F:\AILAB\CF2.PRO   Indent   Ins
predicates
a(integer)
b(integer)
c(integer)
d(integer)
clauses
a(X):-b(X),fail,c(X),!.
a(X):-d(X).
b(1).
b(4).
c(1).
c(3).
d(4).
```

```
Dialog
Yes
Goal: a(X)
X=4
1 Solution
Goal:
```

Program10 for cut predicate.

```
Files   Edit   Run   Compile   Options   Setup
Editor
Line 4   Col 13   F:\AILAB\PRG10.PRO   Indent   I
predicates
        go
clauses
        go:-!,write("102 Rajkot Gujarat"),nl.
```

```
Dialog
Yes
Goal: go
102 Rajkot Gujarat
Yes
Goal: _
```

Program11 for fail predicate.

```
Files   Edit   Run   Compile   Options   Setup
Editor
Line 4   Col 17   F:\AILAB\PRG10.PRO   Indent   I
predicates
        go
clauses
        go:-fail,write("102 Rajkot Gujarat"),nl.
```

```
Dialog
Yes
Goal: go
No
Goal: _
```

Q-6. What are the different types of cut predicates.

Or

Explain red cut and green cut in prolog.

Answer

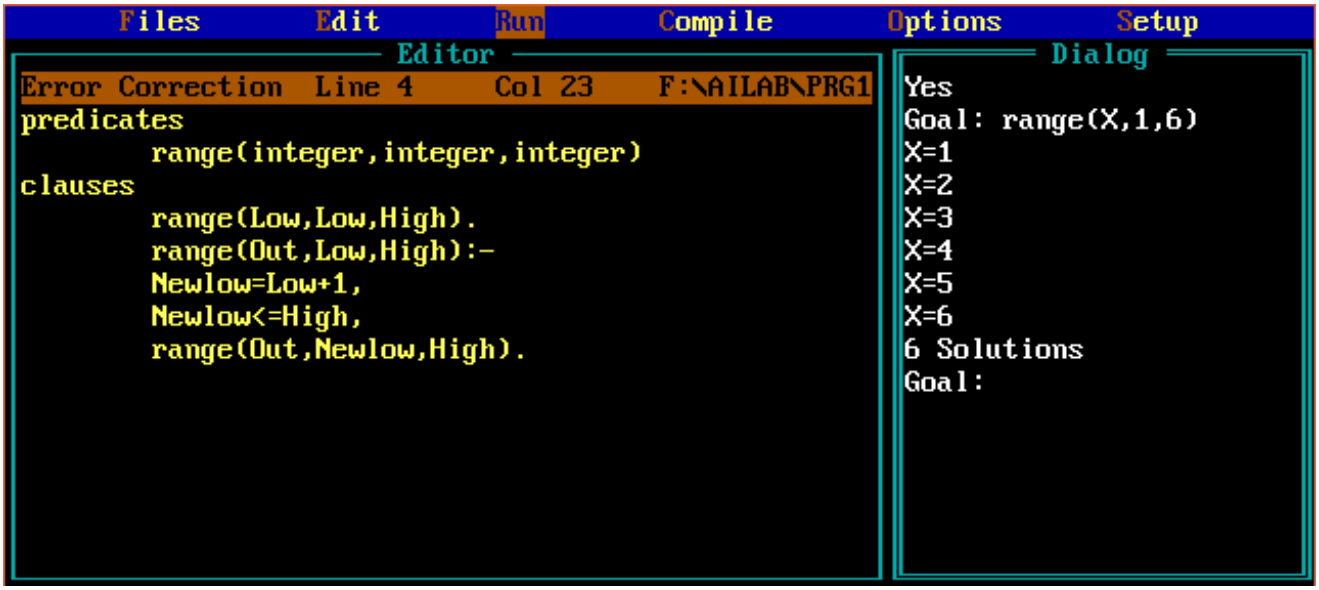
There are two types of cuts. In Prolog, these are called the green and the red cuts.

Green cut: The green type of cut is used to force binding to be retained, once the right clause is reached. Green cuts are used to express determinism. A program is nondeterministic if it is capable of generating multiple solutions on backtracking.

The red type of cut is used to omit explicit conditions. The use of any type of cut in a Prolog program is controversial. It implies a type of procedural control, which is in sharp contrast to the declarative style of Prolog programming. If used with caution, however, cuts improve the clarity and efficiency of most programs.

NOTE: Of the two types of cut, the green cut is the more acceptable type. One can often use the not predicate instead of the red cut.

Program-12.Write a prolog to print the values between given range.



Program-13. Write a prolog program to find minimum no out of 2 numbers.

- **For 5 or 7 mark then write this,**

```
Files    Edit    Run    Compile    Options    Setup
Editor
 Line 1    Col 1    F:\AILAB\PRG13F.PRO   Indent
predicates
        go
        min(integer,integer)
clauses
        go:-
        write("Enter A : "),readint(A),
        write("Enter B : "),readint(B),
        min(A,B).
        min(A,B):-
        A<B,write("A=",A," is minimum."),nl.
        min(A,B):-
        write("B=",B," is minimum."),nl.
```

```
Dialog
Yes
Goal: go
Enter A : 3
Enter B : 6
A=3 is minimum.
Yes
Goal:
```

- **For below 5 mark then write this,**

```
Files    Edit    Run    Compile    Options    Setup
Editor
 Line 7    Col 41    F:\AILAB\PRG13.PRO   Indent   I
predicates
        min(integer,integer)
clauses
        min(A,B):-
        A<B,write("A=",A," is minimum."),nl.
        min(A,B):-
        write("B=",B," is minimum."),nl.
```

```
Dialog
7 is minimum.
Yes
Goal: min(8,7)
7 is minimum.
Yes
Goal: min(7,9)
7 is minimum.
Yes
Goal: min(7,9)
A=7 is minimum.
Yes
Goal: min(9,7)
B=7 is minimum.
Yes
Goal:
```

Program-14. Write a prolog program to find maximum no out of 3 numbers.

⇨ **CODE:**

```
Files        Edit        Run        Compile
 Line 1      Col 1      F:\AILAB\PRG13F.PRO   Indent
predicates
        go
        max(integer,integer,integer)
clauses
        go:-
        write("Enter A : "),readint(A),
        write("Enter B : "),readint(B),
        write("Enter C : "),readint(C),
        max(A,B,C).
        max(A,B,C):-
        A>B,A>C,write("A=",A," is amximum."),nl.
        max(A,B,C):-
        B>A,B>C,write("B=",B," is maximum."),nl.
        max(A,B,C):-
        write("C=",C," is maximum."),nl.
```

OUTPUT: **For 5/7 marks**

```
Files        Edit        Run
C=45 is maximum.
Yes
Goal: go
Enter A : 34
Enter B : 21
Enter C : 12
A=34 is amximum.
Yes
Goal: go
Enter A : 12
Enter B : 34
Enter C : 21
B=34 is maximum.
Yes
Goal:
```

⇨ **For below 5 marks**

```
Files        Edit        Run        Compile        Options        Setup
Editor                                                          Dialog
 Line 8      Col 15     F:\AILAB\PRG14.PRO   Indent   I
predicates                                                      Yes
        max(integer,integer,integer)                            Goal: max(12,21,34)
clauses                                                         C=34 is maximum.
        max(A,B,C):-                                            Yes
        A>B,A>C,write("A=",A," is amximum."),nl.                Goal: max(34,21,12)
        max(A,B,C):-                                            A=34 is amximum.
        B>A,B>C,write("B=",B," is maximum."),nl.                Yes
        max(A,B,C):-                                            Goal: max(12,34,21)
        write("C=",C," is maximum."),nl.                        B=34 is maximum.
                                                                Yes
                                                                Goal:
```

Program-15. Write a prolog program to find sum of 3 numbers.

⇨ **For 5 or 7 mark then write this,**

```
Files    Edit    Run    Compile    Options    Setup
Editor                                          Dialog
 Line 2    Col 11    F:\AILAB\PRG15F.PRO   Indent   Yes
predicates                                       Goal: sum(2,5,7)
        go                                       Sum is 14
        sum(integer,integer,integer)             Yes
clauses                                          Goal: go
        go:-                                     Enter A : 2
        write("Enter A : "),readint(A),          Enter B : 5
        write("Enter B : "),readint(B),          Enter C : 7
        write("Enter C : "),readint(C),          Sum is 14
        sum(A,B,C).                              Yes
        sum(A,B,C):-                             Goal:
        D=A+B+C,write("Sum is ",D),nl.
```

⇨ **For below 5 mark then write this,**

```
Files    Edit    Run    Compile    Options    Setup
Editor                                          Dialog
 Line 5    Col 31    F:\AILAB\PRG15.PRO   Indent   I   Yes
predicates                                       Goal: sum(2,5,7)
        sum(integer,integer,integer)             Sum is 14
clauses                                          Yes
        sum(A,B,C):-                             Goal: _
        D=A+B+C,write("Sum is ",D),nl.
```

Q-7. Define Recursion in prolog.

or

Explain repeat predicate in prolog

Answer

It's a technique of using a clause to invoke a copy of itself.

The following predicate which is not bulletin will be used for recursion.

Repeat ('Repeat')

Syntax:

repeat.

repeat:-

repeat.

This predicate always succeed when it's used in a rule. The repeat predicate is useful for forcing a program to generate alternate solution through backtracking(go back to the initial position and recheck which one is better)

Program-16. Demonstrate a prolog program for the use of 'repeat' predicate.

```
Files    Edit    Run    Compile    Options    Setup
Editor
 Line 1    Col 1    F:\AILAB\PRG16.PRO   Indent   I
domains
        X=integer
predicates
        test
        repeat
clauses
        repeat.
        repeat:-
        repeat.
        test:-
            repeat,
            write("Enter a number > 21 to exit:"),
            readint(X),
            X>21,
            write(X," > 21"),nl.
```

```
Dialog
Yes
Goal: test
Enter a number > 21 to e
xit:9
Enter a number > 21 to e
xit:11
Enter a number > 21 to e
xit:22
22 > 21
Yes
Goal: _
```

Program-17. Write a prolog program to write numbers up to given range by use of recursion without using repeat predicate.

```
Files    Edit    Run    Compile    Options    Setup
Editor
 Line 7    Col 16    F:\AILAB\PRG160.PRO   Indent
domains
        N,C=integer
predicates
        count(integer)
clauses
        count(10).
        count(N):-
                write(" ",N),
                C=N+1,
                count(C).
```

```
Dialog
Yes
Goal: count(1)
 1 2 3 4 5 6 7 8 9Yes
Goal:
```

Q-8. Define unwinding.

Answer

If we want to fix the program as previous example by adding new clause and modifying existing clause slightly.

We can compile and execute the above program with the goal count(1).

It defines the terminating condition. The recursion call will unify with the first and terminates from the loop.

Example of writef predicate

⇨ **CODE:**

```
Files    Edit    Run    Compile    Options    Setup
 Line 1    Col 1    F:\AILAB\PRG10_1.PRO   Indent   Insert
domains
        Name=string
        F=real
predicates
        go
clauses
        go:-
        write("Enter string above 7 character : "),
        readln(Name),
        write("Enter float above 2 decimal point digit : "),
        readreal(F),
        writef("Name=%0.5, float=%0.2f",Name,F),nl.
```

⇨ **OUTPUT:**

```
Files    Edit    Run    Compile    Options    Setup
Yes
Goal: go
Enter string above 7 character : abcdefghi
Enter float above 2 decimal point digit : 123564.59630
Name=abcde, float=123564.60
Yes
Goal: _
```

Program-18. Write a prolog program to find whether the given input is a digit, number, uppercase letter, lowercase letter or symbol.

```
predicates
        check(char)
        go
clauses
        go:-
        write("Enter the character:"),
        readchar(N),
        write(N),nl,
        check(N).
        check(N):-
                N>=47,N<=57,
                write(N," is a digit.").
        check(N):-
                N>=65,N<=90,
                write(N," is a uppercase letter.").
        check(N):-
                N>=97,N<=122,
                write(N," is a lowercase letter.").
        check(N):-
                write(N," is a symbol."),nl.
```

```
Goal: go
Enter the character:d
d is a lowercase letter.Yes
Goal: go
Enter the character:D
D is a uppercase letter.Yes
Goal: go
Enter the character:5
5 is a digit.Yes
Goal: go
Enter the character:{
{ is a symbol.
Yes
Goal:
```

Program-19. Write a prolog program whether a particular number is greater than 50 or not.

```
Files    Edit    Run    Compile    Options    Setup
Editor
Line 5    Col 42    F:\AILAB\PRG19.PRO    Indent    I
predicates
        go
clauses
        go:-
        write("Enter number for check >50 :"),
        readint(X),
        X>50,
        write(X," is greater than 50"),nl.

Dialog
Yes
Goal: go
Enter number for check >
50 :45
No
Goal: go
Enter number for check >
50 :58
58 is greater than 50
Yes
Goal:

Message
Compiling F:\AILAB\PRG19.PRO
go
Compiling F:\AILAB\PRG19.PRO
go

Trace

F2-Save  F3-Load  F5-Zoom  F6-Next  F8-Previous goal  Shift-F10-Resize  F10-End
```

Program-20. Write a prolog program to find the answer of power function for a given value.

```
predicates
        go
        power(integer,integer,integer)
clauses
        go:-
                write("Enter base number : "),
                readint(B),
                write("Enter exponent number : "),
                readint(E),
                power(B,E,B).
        power(B,E,P):-
                E=1,
                write("Power is ",P),nl.
        power(B,E,P):-
                E=0,
                write("Power is 1"),nl.
        power(B,E,P):-
                B>1,
                M=B*P,
                N=E-1,
                power(B,N,M).
```

```
Yes
Goal: go
Enter base number : 6
Enter exponent number :
1
Power is 6
Yes
Goal: go
Enter base number : 6
Enter exponent number : 4
Power is 1296
Yes
Goal: go
Enter base number : 6
Enter exponent number : 0
Power is 1
Yes
Goal: go
Enter base number : 6
Enter exponent number : 1
Power is 6
Yes
Goal:
```

Program-21. Write a prolog program to find a factorial of given number.

```
Files    Edit    Run    Compile    Options    Setup
Editor
 Line 12    Col 9    F:\AILAB\PRG21.PRO   Indent   I
predicates
        fact(integer,integer)
        fact(integer)
clauses
        fact(0,X):-
                X=1.
        fact(N,X):-
                NN=N-1,
                fact(NN,X1),
                X=X1*N.
        fact(N):-
        fact(N,X),
        write("Factorial of ",N," is ",X),nl.
```

```
Dialog
Yes
Goal: fact(5)
Factorial of 5 is 120
Yes
Goal: fact(0)
Factorial of 0 is 1
Yes
Goal: fact(1)
Factorial of 1 is 1
Yes
Goal: fact(3)
Factorial of 3 is 6
Yes
Goal: _
```

- **By 3 varialbles**

```
Files    Edit    Run    Compile    Options    Setup
Editor
 Line 9    Col 31    F:\AILAB\PRG21_1.PRO   Indent
predicates
        fact(integer,integer,integer)
clauses
        fact(0,X,X).
        fact(N,A,X):-
                N>0,
                A1=N*A,
                N1=N-1,
                fact(N1,A1,X).
```

```
Dialog
Yes
Goal: fact(0,1,X)
X=1
1 Solution
Goal: fact(1,1,X)
X=1
1 Solution
Goal: fact(5,1,X)
X=120
1 Solution
Goal: _
```

Program-22. Write a prolog program to login with one username and password.

```
Files    Edit    Run    Compile    Options    Setup
Editor
Error Correction  Line 8    Col 17    F:\AILAB\PRG2
predicates
        login(string,string)
clauses
        login(U,PW):-
                U="abc",
                PW="abc",
                write("Login successful."),nl.
        login(U,PW):-
                write("Login unsuccessful."),nl.
```

```
Dialog
Login successful.
Yes
Goal: login(abc,abc)
Login successful.
Yes
Goal: login(acd,acd)
Login unsuccessful.
Yes
Goal: login(a,ad)
Login unsuccessful.
Yes
Goal: login(abc,abc)
Login successful.
Yes
Goal:
```

Q-9. Define Compound object.

Answer

We can create one object that contains another object. The resulting structure is called compound object.

Ex. Address(Name,Street,City,State,Zip)
The entire object can be treated as single object in predicate.
The first part of the compound object is the object's name which is called functor.
The second part which is a argument list is called component.

Program-23. Write a prolog to demonstrate compound object.

```
Files    Edit    Run    Compile    Options    Setup
Editor
  Line 7    Col 9    F:\AILAB\PRG23_E.PRO    Indent
domains
  things=car(brand);flat;villa;networth(integer)
  name,brand=symbol
predicates
  owns(name,things)
clauses
        owns(jeff,car(bmw)).
        owns(jeff,villa).
        owns(jeff,networth(43)).
        owns(bill,car(audi)).
        owns(bill,flat).
        owns(bill,networth(40)).
```

```
Dialog
Yes
Goal: owns(jeff,Thing)
Thing=car("bmw")
Thing=villa
Thing=networth(43)
3 Solutions
Goal: owns(steve,Thing)
No Solution
Goal: owns(bill,Thing)
Thing=car("audi")
Thing=flat
Thing=networth(40)
3 Solutions
Goal: _
```

```
domains
   symptom=fever(description);cough(type);headache
   description,headache,type,patient=symbol
predicates
   hypothesis(patient,symptom)
clauses
   hypothesis(charlie,fever(high)).
   hypothesis(charlie,cough(high)).
   hypothesis(charlie,headache).
   hypothesis(john,cough(medium)).
   hypothesis(john,fever(low)).
```

```
Yes
Goal: hypothesis(john,Symptom)
Symptom=cough("medium")
Symptom=fever("low")
2 Solutions
Goal: hypothesis(jack,Symptom)
No Solution
Goal: hypothesis(charlie,Symptom)
Symptom=fever("high")
Symptom=cough("high")
Symptom=headache
3 Solutions
Goal: _
```

Q-10. Define String in prolog.

Answer

A string is a list of character.

The properties of a string are as follows:

- All the character in the string must be same type.
- The length of the string can be of length.
- The length of the string can be of length.
- The order of the characters in a string is significant. Ex. “XYZ” is not the same string as “YXZ”.
- The string can be empty or null for Ex. write(“ ”)

Q-11. List out the various operations of string.

Answer

➔ **Concatenation** – The concat is a built-in predicate which will join 2 strings and will form 3rd one.
Syntax: concat(String 1, String 2, ResultString)

Program-24. Write a prolog program which will demonstrate the concatenation operation for 2 strings.

```
Editor
 Line 8     Col 22    F:\AILAB\PRG24.PRO   Indent   I
predicates
        go
clauses
        go:-
            X="Hello world!",
            Y=" Prolog",
            concat(X,Y,Z),
            write(Z),nl.
```

```
Dialog
Yes
Goal: go
Hello world! Prolog
Yes
Goal: _
```

➔ **frontstr predicate** – This predicate will extract the number of characters from the front of a string.

Syntax: frontstr(NumberofCharacters, InputString, SelectedString, RestofString)

NumberofCharacters – Number of left characters to be extracted(Integer)
InputString – input string of charaters
SelectedString – extracted string
RestofString – rest of the input string after extraction process

Program-25. Write a program which will demonstrate frontstr predicate.

```
Editor
 Line 6      Col 24     F:\AILAB\PRG25.PRO   Indent   I
predicates
        go
clauses
        go:-
            X="Hello world! Prolog",
            frontstr(12,X,Select,Rest),
            write(Select),nl,
            write(Rest),nl.
```

```
Dialog
Yes
Goal: go
Hello
world! Prolog
Yes
Goal: go
Hello world!
 Prolog
Yes
Goal: _
```

➔ **fronttoken predicate –** This predicate permits the extraction of token from a string the token can be a name, a number or any non-space character in a sentence each word is token.

Syntax: fronttoken(InputString, Token, RestofString)

Where, InputString – any input string
Token – the first token in the input string(it can be a symbol or a string)
Restofstring – the input string after the token is extended

Program-26. Write a prolog program which will demonstrate the front token predicate.

```
Editor
 Line 6      Col 24     F:\AILAB\PRG25.PRO   Indent   I
predicates
        go
clauses
        go:-
            X="Hello world! Prolog",
            fronttoken(X,Select,Rest),
            write(Select),nl,
            write(Rest),nl.
```

```
Dialog
Yes
Goal: go
Hello
 world! Prolog
Yes
Goal: _
```

➔ **str_len** – It's used for to obtain a length of given string.

Syntax : str_len(InputString,len)

Where, InputString – any input string

Len – length of the InputString(in integer)

Program-27. Write a prolog program which will demonstrate the length of given string.

```
Files    Edit    Run    Compile    Options    Setup
Editor
 Line 5    Col 35    F:\AILAB\PRG27.PRO    Indent    I
predicates
        go
clauses
        go:-
            X="Hello world! Prolog",
            str_len(X,Length),
            write(Length),nl.
```

```
Dialog
Yes
Goal: go
19
Yes
Goal:
```

➔ **upper_lower predicate** – this predicate can be use to convert uppercase characters to lowercase characters or lowercase characters to uppercase characters.

Syntax:upper_lower(UppercaseString,LowercaseString)

Program-28. Write a prolog program to convert lowercase to uppercase.

```
Files    Edit    Run    Compile    Options    Setup
Editor
 Line 5    Col 29    F:\AILAB\PRG28.PRO    Indent    I
predicates
        go
clauses
        go:-
            X="turbo PROLOG2",
            upper_lower(Upper,X),
            write(Upper),nl,
            upper_lower(X,Lower),
            write(Lower),nl.
```

```
Dialog
Yes
Goal: go
TURBO PROLOG2
turbo prolog2
Yes
Goal: _
```

➔ **isname predicate** – this predicate is used to test whether a string is a name or not.
Syntax : isname(String)

Program-29. Write a prolog program for isname predicate.

```
Editor
 Line 1      Col 1      F:\AILAB\PRG29.PRO   Indent   I
predicates
        go
        goh
clauses
        go:-
            X="John",
            isname(X),write("Yes it is name.").
        goh:-
            X="John d",
            isname(X).
```

```
Dialog
Goal: go
Yes it is name.Yes
Goal: goh
No
Goal: _
```

Q-12. What is list in the prolog?

Answer

- A list is an ordered sequence of terms.
- The terms of list can be variables, simple object, compound object or any other list. So, a list can contain unlimited number of terms.
- Each list is set of brackets.
- With the components of the list separated by commas for Ex. [a,b,c,d,e]
- The component of a list should be same domain type it can be integers, real numbers, string, single or compound object.
- We can also indicate empty list as [].

Program-30. Write a prolog program which will demonstrate the declaring of list domain.

```
Files      Edit      Run      Compile      Options      Setup
Editor
 Line 7      Col 48      F:\AILAB\PRG30.PRO   Indent   I
domains
        namelist=names*
        names=symbol
predicates
        club_name(namelist)
clauses
        club_name([android,webbers,design,ui]).
```

```
Dialog
Yes
Goal: club_name([A,B,C,D
])
A=android, B=webbers, C=
design, D=ui
1 Solution
Goal: club_name([A,B])
No Solution
Goal: _
```

Program-31. Write a prolog program which will demonstrate writing a list.

```
Files    Edit    Run    Compile    Options    Setup
Editor
Line 9    Col 32    F:\AILAB\PRG31.PRO    Indent    I
        namelist=symbol*
predicates
        writelist(namelist)
clauses
        writelist([]).
        writelist([Head|Tail]):-
                write(Head),nl,
                writelist(Tail).
```

```
Dialog
all,car])
apple
ball
car



Yes
Goal: writelist([apple,b
all,car])
apple
ball
car
Yes
Goal:
```

Program-32. Write a prolog program which will perform append or concat element in list.

```
Files    Edit    Run    Compile    Options    Setup
Editor
Line 1    Col 1    F:\AILAB\EXP17_2.PRO    Indent
domains
        list=symbol*
predicates
        append(list,list,list)
clauses
        append([],ListB,ListB).
        append([X|List1],List2,[X|List3]):-
                append(List1,List2,List3).
```

```
Dialog
Yes
Goal: append([a,b],[c,d]
,X)
X=["a","b","c","d"]
1 Solution
Goal: _
```

Program-33. Write a prolog program which will perform reverse operation in list.

```
Files    Edit    Run    Compile    Options    Setup
Editor
Line 1    Col 1    F:\AILAB\REVLIST.PRO    Indent
domains
        list=integer*
predicates
        rev_list(list,list)
        rev(list,list,list)
clauses
        rev_list(L1,L2):-
                rev(L1,[],L2).
        rev([],L2,L2).
        rev([H|T],IP,OP):-
                rev(T,[H|IP],OP).
```

```
Dialog

Goal: rev_list([1,2,3,4]
,X)
X=[4,3,2,1]
1 Solution
Goal: rev_list([2,5,3,7]
,X)
X=[7,3,5,2]
1 Solution
Goal: _
```

Program-34. Prolog Program for Tower of Hanoi.

```
Files    Edit    Run    Compile    Options
Line 1    Col 1    F:\AILAB\EXP12.PRO    Indent    Insert
predicates
        hanoi(integer)
        hanoi1(integer,symbol,symbol,symbol)
clauses
        hanoi(P):-
                hanoi1(P,"A","B","C").
        hanoi1(0,R,S,T).
        hanoi1(P,R,S,T):-
                U=P-1,
                hanoi1(U,R,T,S),
                write("Move disk no ",P," from ",R," to ",T),nl,
                readchar(A),
                hanoi1(U,S,R,T).
```

```
Files    Edit
Yes
Goal: hanoi(3)
Move disk no 1 from A to C
Move disk no 2 from A to B
Move disk no 1 from C to B
Move disk no 3 from A to C
Move disk no 1 from B to A
Move disk no 2 from B to C
Move disk no 1 from A to C
Yes
Goal: hanoi(2)
Move disk no 1 from A to B
Move disk no 2 from A to C
Move disk no 1 from B to C
Yes
Goal: _
```

Program-35. Prolog Program for find vowel from list.

```
Files    Edit    Run    C
Line 1    Col 1    F:\AILAB\EXP13.PRO
domains
        list=symbol*
        X=symbol
predicates
        member(X,list)
        vowel(X)
        findvowel(list,integer)
clauses
        member(X,[X|_]).
        member(X,[_|Xs]):-
                member(X,Xs).
        vowel(X):-member(X,[a,e,i,o,u]).
        findvowel([],0).
        findvowel([X|T],N):-
                vowel(X),
                findvowel(T,N1),
                N=N1+1,!.
        findvowel([X|T],N):-
                findvowel(T,N).
```

```
Files    Edit    Run
Yes
Goal: findvowel([a,b,c,d],C)
C=1
1 Solution
Goal: findvowel([b,c,d],C)
C=0
1 Solution
Goal: findvowel([a,b,i,c,e,d,o],C)
C=4
1 Solution
Goal: findvowel([a,i,e,o,u],C)
C=5
1 Solution
Goal:
```

Program-36. Prolog program for find sum of integer list.

```
Files        Edit        Run        Compile
Line 1     Col 1     F:\AILAB\EXP14.PRO   Indent
domains
        list=integer*
predicates
        findsum(list)
        sum(list,integer)
clauses
        findsum(L):-
        sum(L,Sum),
        write("\n Sum of given list : ",Sum),nl.

        sum([],0).

        sum([X|Tail],Sum):-
                sum(Tail,Temp),
                Sum=Temp+X.
```

```
Files        Edit
Yes
Goal: sum([1,2,4,7],S)
S=14
1 Solution
Goal: sum([],S)
S=0
1 Solution
Goal: sum([0,1],S)
S=1
1 Solution
Goal: sum([10,13,14],S)
S=37
1 Solution
Goal:
```

Program-37. Prolog program for check given character is a member of list or not?

```
Files        Edit        Run        Compile        Options        Setup
Editor
Line 1     Col 1     F:\AILAB\EXP17_1.PRO   Indent
domains
        list=symbol*
        X=symbol
predicates
        member(X,list).
clauses
        member(X,[X|_]).
        member(X,[_|Xs]):-
        member(X,Xs).
```

```
Dialog
Yes
Goal: member(a,[a,b,c])
Yes
Goal: member(s,[a,b,c])
No
Goal: member(s,[a,s,c])
Yes
Goal:
```

Program-38. Prolog program for delete an element from list.

```
Files    Edit    Run    Compile    Options    Setup
Editor
 Line 1    Col 1    F:\AILAB\EXP17_3.PRO   Indent
domains
        list=symbol*
predicates
        delete(symbol,list,list)
clauses
        delete(X,[X|Tail],Tail).
        delete(X,[Y|Tail],[Y|Tail1]):-
                delete(X,Tail,Tail1).
```

```
Dialog
Yes
Goal: delete(a,[c,d,a],N
)
N=["c","d"]
1 Solution
Goal: delete(s,[c,d,a],N
)
No Solution
Goal: _
```

Program-39. Prolog program for find maximum from integer list.

```
Files    Edit    Run    Compile    Options    Setup
Editor
 Line 14    Col 17    F:\AILAB\EXP18_1.PRO   Indent
domains
        list=integer*
predicates
        maxL(list,integer)
        max(integer,integer,integer)
clauses
        max(X,Y,X):-
                X>=Y.
        max(X,Y,Y):-
                X<Y.
        maxL([X],X).
        maxL([H|T],X):-
                maxL(T,I),
                max(H,I,X).
```

```
Dialog
Yes
Goal: maxL([2,1,4,3,7],X
)
X=7
1 Solution
Goal: maxL([],X)
No Solution
Goal: maxL([1,10,9],X)
X=10
1 Solution
Goal:
```