



**Atma Ram Sanatan Dharma College**

---

**Artificial Intelligence Practical**  
**File**

---

**SUBMITTED BY**

Name : vaivhav utreja  
Course : Bsc.(Hons) Computer Science  
Roll no : 22/28050  
Semester : 6  
Subject : Artificial Intelligence  
Teacher : Dr. Parul Jain

# Program 1

Write a PROLOG program to implement the family tree and demonstrate the family relationship.

## CODE

```
Program x +
1 % Family tree representation
2 parent(john, mary).
3 parent(john, peter).
4 parent(mary, alice).
5 parent(peter, bob).
6 parent(robert, john).
7 parent(robert, linda).
8 parent(susan, mary).
9 parent(susan, peter).
10
11 male(john).
12 male(peter).
13 male(bob).
14 male(robert).
15
16 female(mary).
17 female(alice).
18 female(linda).
19 female(susan).
20
21 father(X, Y) :- parent(X, Y), male(X).
22 mother(X, Y) :- parent(X, Y), female(X).
23 sibling(X, Y) :- parent(Z, X), parent(Z, Y), X \= Y.
24
25 grandfather(X, Y) :- parent(X, Z), parent(Z, Y), male(X).
26 grandmother(X, Y) :- parent(X, Z), parent(Z, Y), female(X).
27 ancestor(X, Y) :- parent(X, Y).
28 ancestor(X, Y) :- parent(X, Z), ancestor(Z, Y).
29
30 aunt(X, Y) :- sibling(X, Z), parent(Z, Y), female(X).
31 uncle(X, Y) :- sibling(X, Z), parent(Z, Y), male(X).
```

## Output

 *grandfather*(X, bob).

X = john

Next	10	100	1,000	Stop
------	----	-----	-------	------

 *grandmother*(Y, alice).

Y = susan

Next	10	100	1,000	Stop
------	----	-----	-------	------

 *ancestor*(A, bob).

A = peter

Next	10	100	1,000	Stop
------	----	-----	-------	------

## Program 2

Write a PROLOG program to implement `conc(L1, L2, L3)` where L2 is the list to be appended with L1 to get the resulted list L3.

CODE



```
1 % Q2 concatenation of lists
2 conc([],L,L).
3 conc([H|T],L,[H|R]):- conc(T,L,R).
```

Output

```
conc([1,2,3,4],[5,6],R).
```

```
R = [1, 2, 3, 4, 5, 6]
```

```
?- conc([1,2,3,4],[5,6],R).
```

## Program 3


Write a PROLOG program to implement reverse(L, R) where List L is original and List R is reversed list.

CODE



```
1 % Q3 Reversing a List
2 conc([],L,L).
3 conc([H|T],L,[H|R]):- conc(T,L,R).
4
5 reverse([],[]).
6 reverse([H|T],R):-
7     reverse(T,RevT),conc(RevT,[H],R).
```

## Output

 `reverse([1,2,3,4],Rev).`

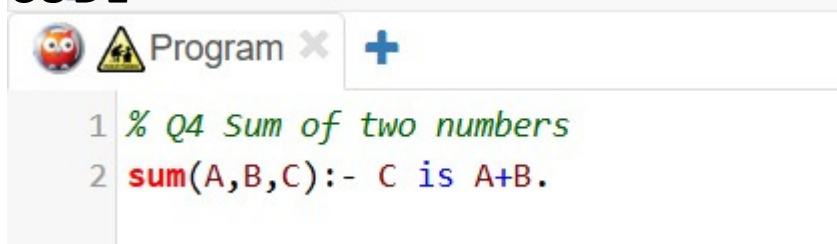
**Rev** = [4, 3, 2, 1]

?- `reverse([1,2,3,4],Rev).`

## Program 4

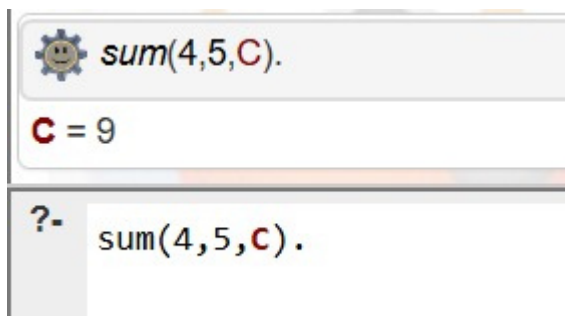
Write a PROLOG program to calculate the sum of two numbers.

CODE



```
1 % Q4 Sum of two numbers
2 sum(A,B,C):- C is A+B.
```

Output



```
?- sum(4,5,C).
C = 9
```

## Program 5

Write a PROLOG program to implement  $\text{max}(X, Y, M)$  so that  $M$  is the maximum of two numbers  $X$  and  $Y$ .

### CODE



Program ✕



```
1 % Maximum of two numbers
2 max(X, Y, X) :- X >= Y.
3 max(X, Y, Y) :- Y > X.
```

### Output



$\text{max}(5, 10, S).$

**S** = 10

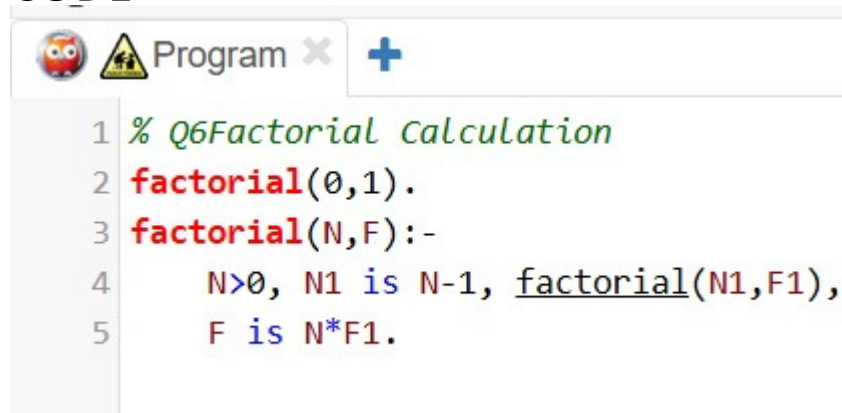
?-

$\text{max}(5, 10, S).$

## Program 6

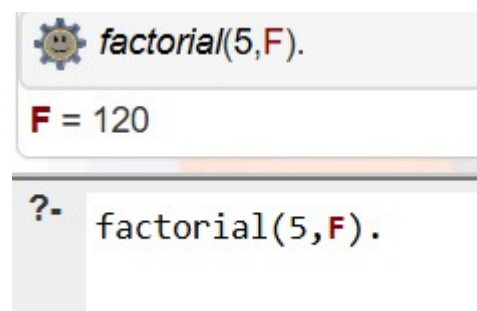
Write a program in PROLOG to implement factorial (N, F) where F represents the factorial of a number N.

### CODE



```
1 % Q6Factorial calculation
2 factorial(0,1).
3 factorial(N,F):-
4     N>0, N1 is N-1, factorial(N1,F1),
5     F is N*F1.
```

### Output



```
factorial(5,F).
F = 120
?- factorial(5,F).
```



## Program 7

Write a program in PROLOG to implement generate\_fib(N,T) where T represents the Nth term of the Fibonacci series.

### CODE

```
Program x +
1 % Q7 fibonacci number generation
2 fib(0,0).
3 fib(1,1).
4 fib(N,T):-
5     N>1,N1 is N-1,N2 is N-2,
6     fib(N1,T1),fib(N2,T2),T is T1+T2.
7
```

### Output

```
fib(6,T).
T = 8
?- fib(6,T).
```

## Program 8

Write a PROLOG program to implement power (Num, Pow, Ans) : where Num is raised to the power Pow to get Ans.

### CODE

```
1 % Q8 Power function
2 power(_,0,1).
3 power(Num,Pow,Ans):-
4     Pow > 0,Pow1 is Pow-1,
5     power(Num,Pow1,Ans1),
6     Ans is Num*Ans1.
7
```

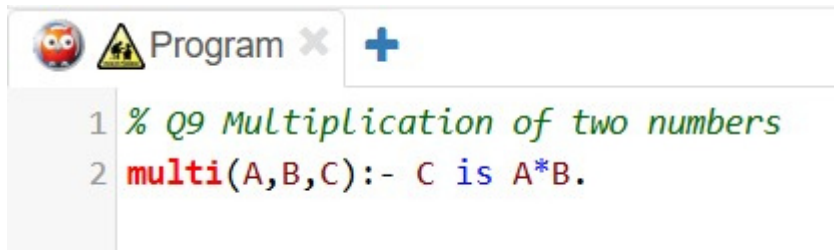
### Output

```
power(2,5,S).
S = 32
?- power(2,5,S).
```

## Program 9

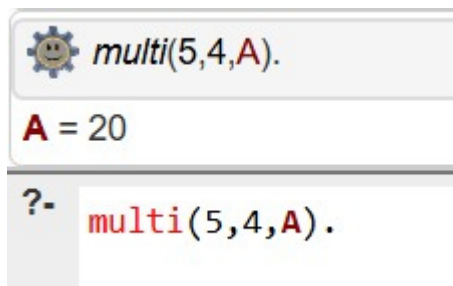
PROLOG program to implement multi (N1, N2, R) :  
where N1 and N2 denotes the numbers to be multiplied  
and R represents the result.

### CODE



```
1 % Q9 Multiplication of two numbers
2 multi(A,B,C):- C is A*B.
```

### Output



```
?- multi(5,4,A).
A = 20
```

## Program 10

Write a PROLOG program to implement `memb(X, L)`: to check whether X is a member of L or not.

### CODE



```
1
2 % Q10 check membership in a List
3 memb(X,[X|_]).
4 memb(X,[_|T]):- memb(X,T).
5
```

### Output

 `memb(4,[1,2,3,5,5]).`

**false**

 `memb(1,[1,2,3,5,5]).`


**true**

?- `memb(4,[1,2,3,5,5]).`  
`memb(1,[1,2,3,4]).`

## Program 11

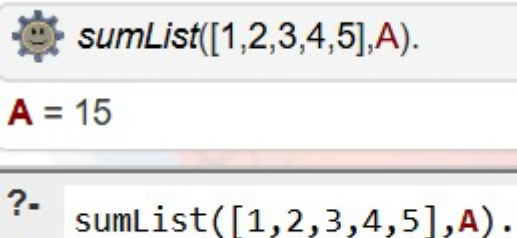
Write a PROLOG program to implement `sumlist(L, S)` so that `S` is the sum of a given list `L`.

### CODE



```
1 % Q11 sum of a list
2 sumList([],0).
3 sumList([H|T],S):-
4     sumList(T,S1), S is H+S1.
```

### OUTPUT



```
sumList([1,2,3,4,5],A).
A = 15
?- sumList([1,2,3,4,5],A).
```




## Program 12

Write a PROLOG program to implement two predicates `evenlength(List)` and `oddlength(List)` so that they are true if their argument is a list of even or odd length respectively.

### CODE

```
Program x +
1 % Q12 even and odd length of the list
2 evenLength([]).
3 evenLength(_|_|List):-
4     evenLength(List).
5 oddLength(_).
6 oddLength(_|_|List):-
7     oddLength(List).
```

### OUTPUT

 <code>evenLength([1,2,3,4]).</code> <b>true</b>	 <code>evenLength([2,3,4]).</code> <b>false</b>
?- <code>evenLength([1,2,3,4]).</code>	?- <code>evenLength([2,3,4]).</code>
 <code>oddLength([2,3,4]).</code> <b>true</b>	
?- <code>oddLength([2,3,4]).</code>	

## Program 13

Write a PROLOG program to implement maxlist(L, M) so that M is the maximum number in the list.

### CODE



```
1 % Q13 Maximum element in the List
2 maxList([M],M).
3 maxList([H|T],M):-
4     maxList(T,M1),
5     (H > M1 -> M=H; M=M1).
```

### OUTPUT

```
maxList([4,5,10,11],S).
```

**S** = 11

?- maxList([4,5,10,11],**S**).

## Program 14

Write a PROLOG program to implement insert(I, N, L, R) that inserts an item I into Nth position of list L to generate a list R.

### CODE

```
Program x +
1 % Insert element at Nth position
2 insert(I, 1, L, [I|L]).
3 insert(I, N, [H|T], [H|R]) :-
4     N > 1, N1 is N - 1,
5     insert(I, N1, T, R).
```

### OUTPUT

```
insert(2,3,[4,6,8,10],R).
R = [4, 6, 2, 8, 10]
?- insert(2,3,[4,6,8,10],R).
```



## Program 15

Write a PROLOG program to implement delete(N, L, R) that removes the element on Nth position from a list L to generate a list R.

### CODE

```
Program x +
1 % Q15 Delete element at Nth Position
2 delete(1,[_|T],T).
3 delete(N,[H|T],[H|R]):-
4     N > 1 , N1 is N-1, delete(N1,T,R).
5
```

### OUTPUT

```
delete(2,[4,6,8,10],R).
R = [4, 8, 10]
?- delete(2,[4,6,8,10],R).
```