
# Project File
# AI and Knowledge Representation

By: Tushar Gupta
Btech CSE
Semester 6
210BTCSECS014

Submitted to:
Mrs. Meenakshi Gupta

# INDEX

| | | | |
|---|---|---|---|
| 12 | Write a program in prolog to find the intersection of two lists entered. | 13 | |
| 13 | Write a program in prolog to find if the entered element is maximum or not. | 14 | |
| 14 | Write a program in prolog to find the sum of entered elements of list. | 15 | |

**<u>Code 1</u>**- *Write a program in Prolog to create a knowledge Base 1 and query it.*

```
girl(priya).
girl(seema).
boy(rahul).

can_cook(priya).
can_sing(rahul).
```

```
| ?- girl(priya)
.

yes
| ?- girl(seema).

yes
| ?- can_cook(priya)
.

yes
| ?- can_sing(priya).

no
| ?-
```

**Code 2**- *Write a program in Prolog to create KB with rules and query it.*

```
girl(mary).
girl(susie).
boy(john).
boy(peter).

likes(john,mary).
likes(john,susie).
likes(peter,john).
likes(susie,peter).
hates(john,peter).
likes(john,jane).

friend_of(john,peter):-likes(john,peter).
friend_of(john,susie):-likes(peter,susie).
friend_of(john,mary):-likes(john,peter),likes(peter,susie).
friend_of(john,mary):-likes(john,peter);likes(peter,susie).

friend_of(john,jane):-likes(john,jane).
```

```
| ?- friend_of(john,peter).

no
| ?- friend_of(john,susie).

no
| ?- friend_of(john,jane).

no
| ?- friend_of(john,mary).

no
```

**Code 3**- *Write a program in prolog to create KB of countries and cities and create rules for a city located in a specific country. Query this program.*

```
located_in(atlanta,georgia).
located_in(houston,texas).
located_in(austin,texas).
located_in(toronto,ontario).
located_in(X,usa):-located_in(X,texas).
located_in(X,usa):-located_in(X,georgia).
located_in(X,canada):-located_in(X,ontario).
located_in(X,north_america):-located_in(X,usa)
located_in(X,north_america):-located_in(X,canada)
```

```
| ?- located_in(atlanta,georgia).

true ?

yes
| ?- located_in(atlanta,usa).

true ?

yes
| ?- located_in(atlanta,texas).

no
| ?-
located_in(X,texas).

X = houston ? a

X = austin

no
```

**Code 4**- *Write a program in prolog to create relations in a family tree and query it.*

```
male(jonathan).
male(john).
male(arnold).
male(lousie).
male(kelly).

female(jennifer).
female(martha).
female(mary).
female(alice).
female(nancy).

parent(jonathan,martha).
parent(jennifer,martha).
parent(jonathan,john).
parent(jennifer,john).
parent(jonathan,arnold).
parent(jennifer,arnold).

parent(john,kelly).
parent(mary,kelly).
parent(john,alice).
parent(mary,alice).

parent(arnold,nancy).
parent(arnold,lousie).

mother(X,Y):-parent(X,Y),female(X).
father(X,Y):-parent(X,Y),male(X).
brother(X,Y):-parent(Z,X),parent(Z,Y),male(X).
sister(X,Y):-parent(Z,X),parent(Z,Y),female(X).
uncle(X,Y):-parent(Z,Y),brother(X,Z).
aunt(X,Y):-parent(Z,Y),sister(X,Z).
grandfather(X,Y):-parent(Z,Y),father(X,Z).
grandmother(X,Y):-parent(Z,Y),mother(X,Z).
```

```
| ?- mother(X,arnold).

X = jennifer ? .
Action (; for next solution, a for all solutions, RET to stop) ? a

no
| ?- brother(jennifer,X)
.

no
| ?- brother(X,jennifer).

no
| ?- male(X).

X = jonathan ? a

X = john

X = arnold

X = lousie

X = kelly

yes
| ?- brother(X).
uncaught exception: error(existence_error(procedure,brother/1),top_level/0)
| ?- brother(X,arnold).

X = john ? a

X = john

X = arnold

X = arnold

(15 ms) no
| ?- uncle(X,arnold).

no
```

**Code 5**- *Write a program in prolog to find if an element is a member of a list.*

```
member(X,[X|_]).
member(X,[_|T]):-member(X,T).
```

```
| ?- member(3,[1,2,3,4]).

true ? ;

no
| ?- member(5,[1,2,3,4]).

no
```

**Code 6**- *Write a program in prolog to concatenate two lists.*

```
list_concat([],L,L).
list_concat([X1|L1],L2,[X1|L3]) :- list_concat(L1,L2,L3).
```

```
| ?- list_concat([1,2,3],[4,5,6],List).

List = [1,2,3,4,5,6]

yes
| ?- list_concat([1,2,3],[a,b,c],List).

List = [1,2,3,a,b,c]

yes
```

**Code 7**- *Write a program in prolog to delete an element from list.*

```
list_concat([],L,L).
list_concat([X1|L1],L2,[X1|L3]) :- list_concat(L1,L2,L3).
```

```
yes
| ?- list_delete(6,[1,2,3,4,5,6],List).

List = [1,2,3,4,5] ?

yes
| ?- list_delete(X,[1,2,3,4,5,6],[1,2,3,5,6]).

X = 4 ?

yes
```

**Code 8**- *Write a program in prolog to insert an element into list.*

```
list_delete(X,[X],[]).
list_delete(X,[X|L1],L1).
list_delete(X,[Y|L2],[Y|L1]):-list_delete(X,L2,L1).
list_insert(X,L,R):-list_delete(X,R,L).
```

```
(16 ms) yes
| ?- list_insert(6,[1,2,3,4,5],List).

List = [6,1,2,3,4,5] ? a

List = [1,6,2,3,4,5]

List = [1,2,6,3,4,5]

List = [1,2,3,6,4,5]

List = [1,2,3,4,6,5]

List = [1,2,3,4,5,6]

List = [1,2,3,4,5,6]

no
```

**Code 9**- *Write a program in prolog to reverse a list.*

```
list_concat([],L,L).
list_concat([X1|L1],L2,[X1|L3]):-list_concat(L1,L2,L3).

list_rev([],[]).
list_rev([Head|Tail],Reversed):-list_rev(Tail,RevTail),list_concat(RevTail,[Head],Reversed).
```

```
yes
| ?- list_rev([1,2,3,4,5],List).

List = [5,4,3,2,1]

yes
| ?- list_rev([1,2,3,4,5],[5,4,3,2,1]).

yes
```

**Code 10**-*Write a program in prolog to check if the entered list is ordered or not.*

```
list_order([X, Y | Tail]) :- X =< Y, list_order([Y|Tail]).
list_order([X]).
```

```
yes
| ?- list_order([1,2,3,4,5]).

true ?

yes
| ?- list_order([1,3,2,5,4]).

no
| ?-
```

**Code 11**- *Write a program in prolog to find the union of two lists entered.*

```
member(X,[X|_]).
member(X,[_|TAIL]) :- member(X,TAIL).
list_union([X|Y],Z,W) :- member(X,Z),list_union(Y,Z,W).
list_union([X|Y],Z,[X|W]) :- \+member(X,Z),list_union(Y,Z,W).
list_union([],Z,Z).
```

```
yes
| ?- list_union([1,2,3],[1,3,5],List).

List = [2,1,3,5] ?

yes
| ?- list_union([a,b,c],[1,2,3],List).

List = [a,b,c,1,2,3]

yes
```

**Code 12-** *Write a program in prolog to find the intersection of two lists entered.*

```
member(X,[X|_]).
member(X,[_|TAIL]) :- member(X,TAIL).

list_intersect([X|Y],Z,[X|W]) :- member(X,Z), list_intersect(Y,Z,W).
list_intersect([X|Y],Z,W):- \+ member(X,Z),list_intersect(Y,Z,W).
list_intersect([],Z,[]).
```

```
yes
| ?- list_intersect([1,2,3],[1,3,5],List).

List = [1,3] ?

yes
| ?- list_intersect([2,4,6],[1,3,5],List).

List = []

yes
| ?- |
```

**Code 13**- *Write a program in prolog to find if the entered element is maximum or not.*

```
max(X,Y,X) :- X >= Y.
max(X,Y,Y) :- X < Y.
list_max([X],X).
list_max([X,Y|Rest],Max) :- list_max([Y|Rest],MaxRest), max(X,MaxRest,Max).
```

```
yes
| ?- list_max([1,3,6,9],Max).

Max = 9 ?

yes
| ?- list_max([13,36,67,92],Max).

Max = 92 ?

yes
| ?- |
```

**Code 14**- *Write a program in prolog to find the sum of entered elements of list.*

```
list_sum([],0).
list_sum([Head|Tail],Sum) :- list_sum(Tail, SumTemp), Sum is Head + SumTemp.
```

```
yes
| ?- list_sum([2,3,5,6,9],Sum).

Sum = 25

yes
| ?- list_sum([21,34,57,68,90],Sum).

Sum = 270

yes
| ?-
```