

**Jaiwin Shah**

**2019130058**

**Batch C**

**AIML lab**

Aim: Prolog Problem 1(Family tree)

Theory:

Prolog:

Prolog is a logic programming language associated with artificial intelligence and computational linguistics. Prolog has its roots in first-order logic, a formal logic, and unlike many other programming languages, Prolog is intended primarily as a declarative programming language: the program logic is expressed in terms of relations, represented as facts and rules. A computation is initiated by running a query over these relations.

The language was developed and implemented in Marseille, France, in 1972 by Alain Colmerauer with Philippe Roussel, based on Robert Kowalski's procedural interpretation of Horn clauses.

Prolog was one of the first logic programming languages and remains the most popular such language today, with several free and commercial implementations available. The language has been used for theorem proving, expert systems, term rewriting, type systems, and automated planning, as well as its original intended field of use, natural language processing. Modern Prolog environments support the creation of graphical user interfaces, as well as administrative and networked applications. Prolog is well-suited for specific tasks that benefit from rule-based logical queries such as searching databases, voice control systems, and filling templates.


Implementation:

1. Create a family tree using PROLOG. It should have rules for father, mother, brother, sister, grandparent, uncle, aunt, predecessors, successors.


Code:

```
female(arachana).
female(neha).
female(jyoti).
male(jaiwin).
male(shrey).
male(atul).
male(abhay).
male(jolly).
parent(atul,jaiwin).
parent(arachana,jaiwin).
parent(atul,shrey).
parent(arachana,shrey).
parent(jolly,atul).
parent(neha,atul).
parent(jolly,abhay).
parent(neha,sangita).
parent(jolly,jyoti).
parent(neha,jolly).
mother(X,Y):- parent(X,Y),female(X).
father(X,Y):- parent(X,Y),male(X).
son(X,Y):- parent(Y,X),male(X).
daughter(X,Y):- parent(Y,X),female(X).
sister(X,Y):- parent(Z,X),parent(Z,Y),female(X),X\==Y.
brother(X,Y):-parent(Z,X),parent(Z,Y),male(X),X\==Y.
grandparent(X,Y):- parent(X,Z),parent(Z,Y).
successor(X,Y):- parent(Y,X).
successor(X,Y):- parent(Z,X),successor(Z,Y).
predecessor(X,Y):- parent(X,Y).
predecessor(X,Y):- parent(Y,Z),parent(Z,X).
uncle(X,Y):- brother(X,Z),parent(Z,Y).
aunt(X,Y):- sister(X,Z),parent(Z,Y).
```


Output:

 `mother(arachana,jaiwin).`


true

 `father(atul,jaiwin).`

true

 `aunt(arachana,jaiwin).`

false

 `aunt(jyoti,jaiwin).`

true

 `grandparent(jolly,jaiwin).`


true

 `grandparent(neha,jaiwin).`

true

 `predecessor(neha,jaiwin).`


false

 `predecessor(neha,atul).`

true

 `successor(atul,neha).`

true

 `brother(jaiwin,shrey).`

true

2. Given a list [a,a,a,a,b,b,b,c,c]. Write a function that does the following  
`rle([a,a,a,a,b,b,c,c],X)`, X: [a,b,c]

Code:

`rle([],[]). rle([X],[X]).`

`rle([X,X|Xs],Zs) :-`

`rle([X|Xs],Zs).`

`rle([X,Y|Ys],[X|Zs]) :- X \= Y, rle([Y|Ys],Zs).`

 `rle([a,a,a,a,b,b,c,c],X).`

**X** = [a, b, c]

3. Given a list [a,b,c,d,e,f,g]. Write a function that does the following  
 slice([a,b,c,d,e,f,g],[2,5],X)

Code:

```
slice([X|_],1,1,[X]).
slice([X|Xs],1,K,[X|Ys]) :- K > 1,
K1 is K - 1, slice(Xs,1,K1,Ys).
slice([_|Xs],I,K,Ys) :- I > 1,
I1 is I - 1, K1 is K - 1, slice(Xs,I1,K1,Ys).
```

 slice([a,b,c,d,e,f,g],2,5,X).

X = [b, c, d, e]

4. Group list into sublists according to the distribution given For example,  
 subsets([a,b,c,d,e,f,g],[2,2,3],X,[]) should return X = [[a,b][c,d][e,f,g]] The  
 order of the list does not matter

```
Code: el(X,[X|L],L).
el(X,[_|L],R) :- el(X,L,R).
selectN(0,_,[]) :- !.
selectN(N,L,[X|S]) :- N >
0, el(X,L,R), N1 is N-
1, selectN(N1,R,S).
subsets([],[],[],[]).
subsets(G,[N1|Ns],[G1|Gs],[]) :-
selectN(N1,G,G1),
subtract(G,G1,R),
subsets(R,Ns,Gs,[]).
```

Output:

 subsets([a,b,c,d,e,f,g],[2,2,3],X,[]).

X = [[a, b], [c, d], [e, f, g]]

Conclusion:

From this experiment I learned about prolog and implemented the family tree and the given problems using prolog.