Prolog

Samkit Shah
2019130060
Batch C
TE Comps

Aim

To solve problems using the Prolog Programming Language.

Theory Prolog

Prolog is a logic programming language. It has important role in artificial intelligence. Unlike many other programming languages, Prolog is intended primarily as a declarative programming language. In prolog, logic is expressed as relations (called as Facts and Rules). Core heart of prolog lies at the logic being applied. Formulation or Computation is carried out by running a query over these relations In prolog, We declare some facts. These facts constitute the Knowledge Base of the system. We can query against the Knowledge Base. We get output as affirmative if our query is already in the knowledge Base or it is implied by Knowledge Base, otherwise we get output as negative. So, Knowledge Base can be considered similar to database, against which we can query. Prolog facts are expressed in definite pattern. Facts contain entities and their relation. Entities are written within the parenthesis separated by comma (,). Their relation is expressed at the start and outside the parenthesis. Every fact/rule ends with a dot (.).

Problem Statement

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

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]

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)

   X : [c,d,e,f]

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]]

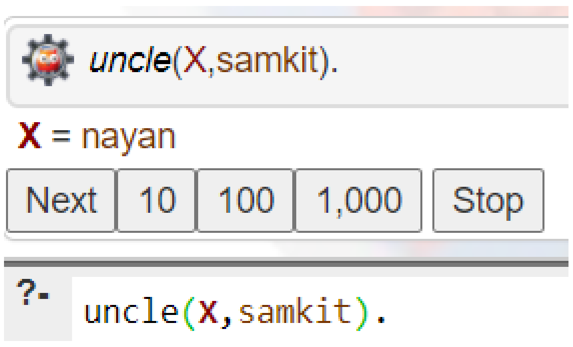
   The order of the list does not matter

Program 1:
Code:

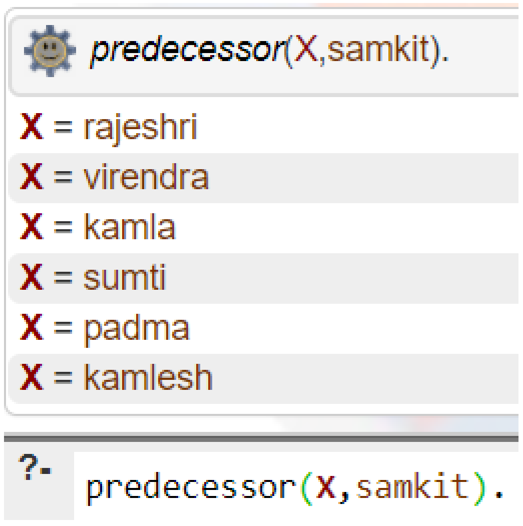
```
parent(rajeshri, aagam).
parent(virendra, aagam).
parent(rajeshri, samkit).
parent(virendra, samkit).
parent(kamla, rajeshri).
parent(sumti,rajeshri).
parent(padma, virendra).
parent(padma, nayan).
parent(kamlesh,virendra).
parent(kamlesh, nayan).

female(rajeshri).
female(kamla).
female(padma).
male(aagam).
male(samkit).
male(virendra).
male(kamlesh).
male(sumti).
male(nayan).

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).

grandfather(X, Y):- parent(X, A), parent(A, Y), male(X).
grandmother(X, Y):- parent(X, A), parent(A, Y), female(X).

sister(X, Y):- parent(A, X), parent(A, Y), female(X), X \= Y.
brother(X, Y):- parent(A, X), parent(A, Y), male(X), X \= Y.

aunt(X, Y):- sister(X, Z), parent(Z, Y).
```

```
uncle(X, Y):- brother(X, Z), parent(Z, Y).

predecessor(X, Y) :- parent(X, Y).
predecessor(X, Y) :- parent(X, A),predecessor(A, Y).

successor(X, Y):- son(Y, X).
successor(X, Y):- daughter(Y, X).
successor(X, Y):- son(A, X), successor(A, Y).
successor(X, Y):- daughter(A, X), successor(A, Y).
```

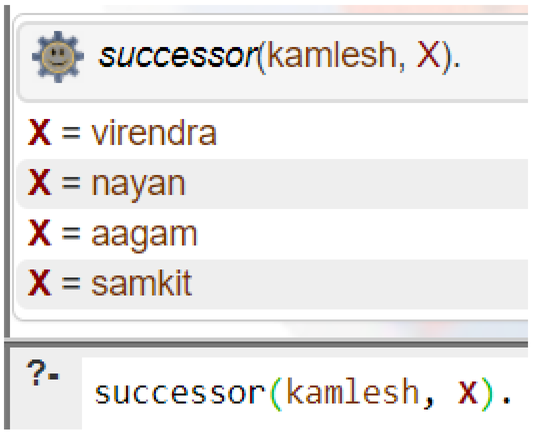
Output:



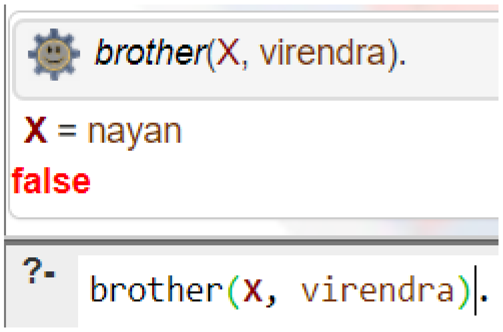
```
uncle(X,samkit).
X = nayan
?- uncle(X,samkit).
```



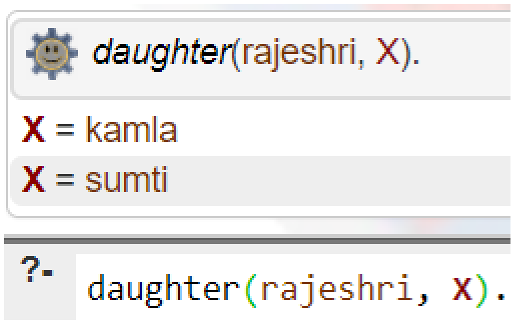
```
predecessor(X,samkit).
X = rajeshri
X = virendra
X = kamla
X = sumti
X = padma
X = kamlesh
?- predecessor(X,samkit).
```



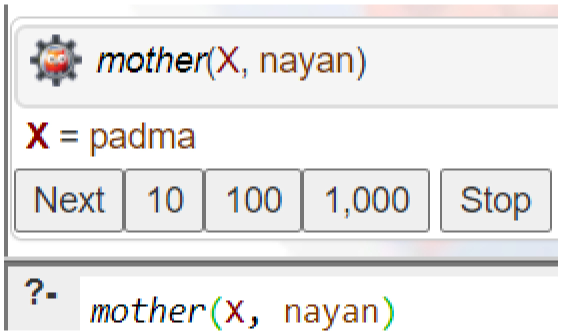
```
successor(kamlesh, X).
X = virendra
X = nayan
X = aagam
X = samkit
?- successor(kamlesh, X).
```



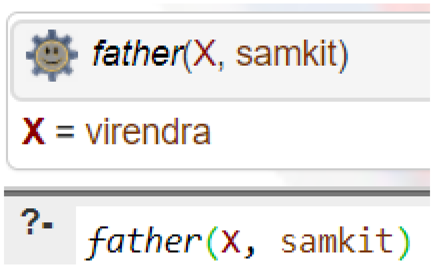
```
brother(X, virendra).
X = nayan
false
?- brother(X, virendra).
```



```
daughter(rajeshri, X).
X = kamla
X = sumti
?- daughter(rajeshri, X).
```



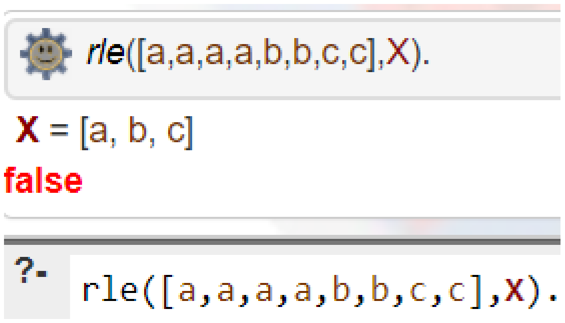
```
mother(X, nayan)
X = padma
?- mother(X, nayan)
```



```
father(X, samkit)
X = virendra
?- father(X, samkit)
```

Program 2:
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).
```



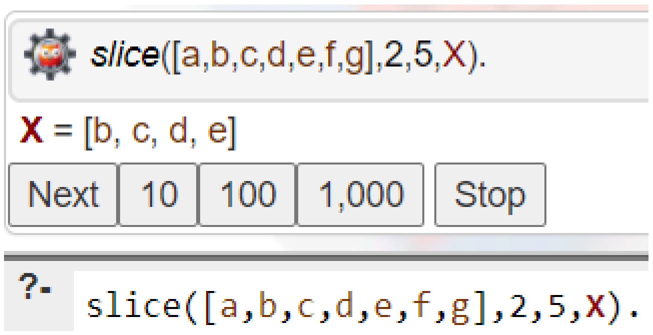
```
rle([a,a,a,a,b,b,c,c],X).
X = [a, b, c]
false
?- rle([a,a,a,a,b,b,c,c],X).
```

Program 3:

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).
```

Output:



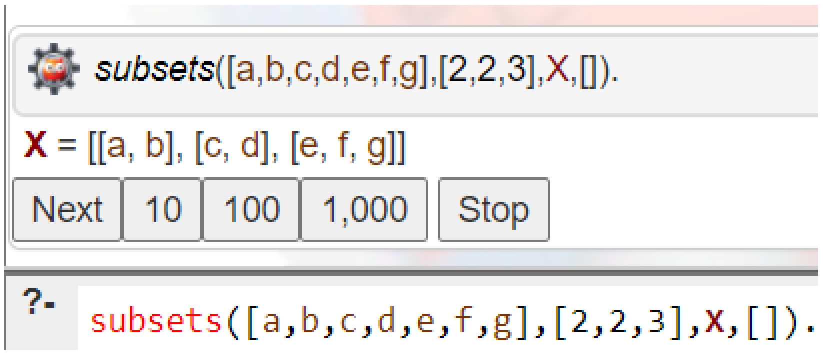
```
slice([a,b,c,d,e,f,g],2,5,X).
X = [b, c, d, e]
```

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

Program 4:
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]]
```

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

Conclusion:

For logical programming, I learned how to use Prolog. This project helped me learn the terminologies as well as the syntax of programming in Prolog. I learned the distinction between facts and relations, as well as many list operations such as concatenation, slicing, and so on.