Name: Apurv Sarode

Subject: AI/ML

UID: 2019130054

Branch: TE Comps

Batch: C

Experiment 6

Aim: Prolog Problems

Code:

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

```
female(vaishali).
female(sheetal).
female(sudarshana).
female(shalini).
female(yutika).
female(chandrabhaga).
male(rajendra).
male(dinesh).
male(apurv).
male(rohit).
male(rihansh).
male(bala).
male(namdeorao).

parent(chandrabhaga,rajendra).
```

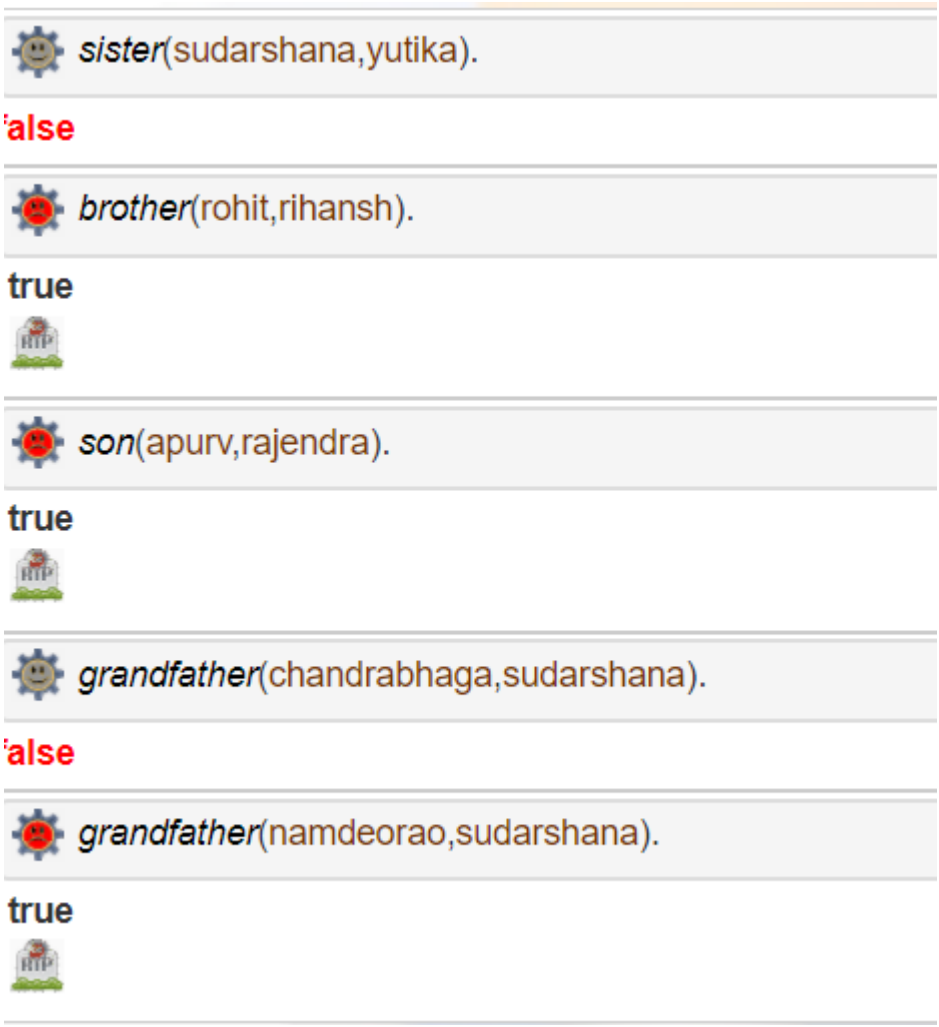
```
parent(chandrabhaga,dinesh).
parent(namdeorao,rajendra).
parent(namdeorao,dinesh).
parent(rajendra,apurv).
parent(rajendra,sudarshana).
parent(vaishali,apurv).
parent(vaishali,sudarshana).
parent(dinesh,rohit).
parent(dinesh,rihansh).
parent(sheetal,rohit).
parent(sheetal,rihansh).
parent(bala,yutika).
parent(shalini,yutika).
father(X,Y) :- male(X),parent(X,Y).
mother(X,Y) :- female(X),parent(X,Y).
son(X,Y) :- male(X),parent(Y,X).
daughter(X,Y) :- female(X),parent(Y,X).
grandfather(X,Y) :- male(X),parent(X,Somebody),parent(Somebody,Y).
grandmother(X,Y) :- female(X),parent(X,Somebody),parent(Somebody,Y).
sister(X,Y) :- female(X),parent(Par,X),parent(Par,Y), X \= Y.
brother(X,Y) :- male(X),parent(Par,X),parent(Par,Y), X \= Y.
aunt(X,Y) :- female(X),sister(X,Mom),mother(Mom,Y).
aunt(X,Y) :- female(X),sister(X,Dad),father(Dad,Y).
uncle(X,Y) :- male(X),brother(X,Mom),mother(Mom,Y).
uncle(X,Y) :- male(X),brother(X,Dad),father(Dad,Y).
cousin(X,Y) :- uncle(Unc,X),father(Unc,Y).
predecessor(X,Y) :- parent(X,Y).
predecessor(X,Y) :- parent(X,Somebody),parent(Somebody,Y).
successor(X,Y) :- son(X,Y).
```

successor(X,Y) :- daughter(X,Y).

successor(X,Y) :- son(Somebody,X),successor(Somebody,Y).

successor(X,Y) :- daughter(Somebody,X),successor(Somebody,Y).

OUTPUT:



Q.2

Given a list [a,a,a,a,b,b,b,c,c]

write a function that does the following

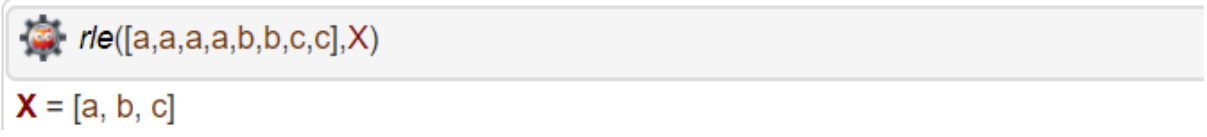
rle([a,a,a,a,b,b,c,c],X)

X: [a,b,c]

Code:

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

rle([X], [X]).

rle([X, X|REMAINING], OUTPUT) :- rle([X|REMAINING], OUTPUT).

rle([X, Y|REMAINING], [X|OUTPUT_TAIL]) :- X\=Y, rle([Y|REMAINING], OUTPUT_TAIL).
```

OUTPUT:



Q.3] Given a list [a,b,c,d,e,f,g]

write a function that does the following

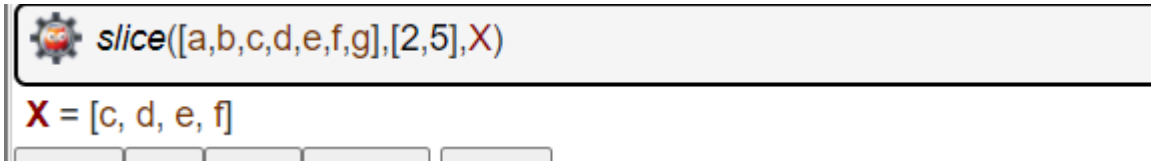
slice([a,b,c,d,e,f,g],[2,5],X)

X: [c,d,e,f]

Code:

```
slice([X|_], 1, 1, [X]).

slice([X|TAIL], 1, CURRENT_INDEX, [X|REM_TAIL]) :- CURRENT_INDEX > 1,

NEXT_INDEX is CURRENT_INDEX - 1, slice(TAIL, 1, NEXT_INDEX, REM_TAIL).

slice([_|TAIL], I, CURRENT_INDEX, OUTPUT) :- I > 1, I1 is I - 1, NEXT_INDEX is CURRENT_INDEX - 1,
slice(TAIL, I1, NEXT_INDEX, OUTPUT).
```

OUTPUT:



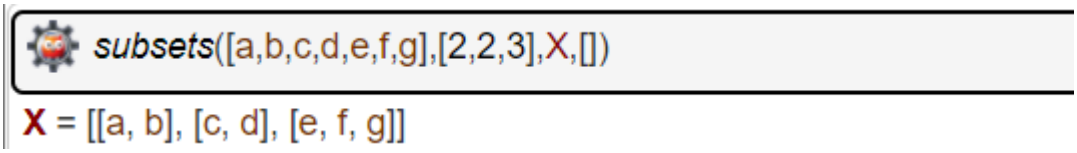
4. Group list into sublists according to the distribution given
For example
subsets([a,b,c,d,e,f,g],[2,2,3],X,[])
should return X = [[a,b][c,d][e,f,g]]
The order of the list does not matter

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:



5. Huffman Code

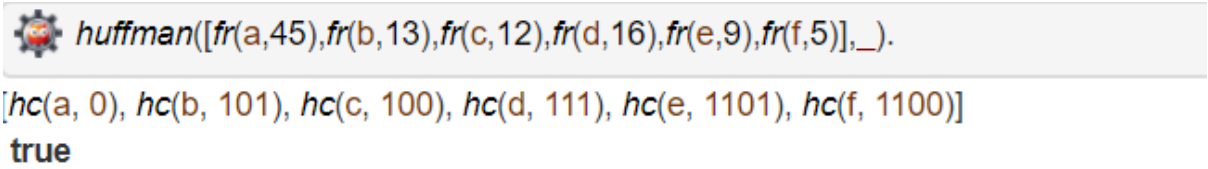
We suppose a set of symbols with their frequencies, given as a list of fr(S,F) terms.
Example:
[fr(a,45),fr(b,13),fr(c,12),fr(d,16),fr(e,9),fr(f,5)].
Our objective is to construct a list hc(S,C) terms,
where C is the Huffman code word for the symbol S.
In our example, the result could be
Hs =[hc(a,'0'), hc(b,'101'), hc(c,'100'), hc(d,'111'), hc(e,'1101'), hc(f,'1100')]
[hc(a,'01'),...etc.].
The task shall be performed by the predicate huffman/2 defined as follows:
% huffman(Fs,Hs) :- Hs is the Huffman code table for the frequency table Fs

Code:

```
huffman(Fs,Cs) :-

initialize(Fs,Ns),

make_tree(Ns,T),

traverse_tree(T,Cs).

initialize(Fs,Ns) :- init(Fs,NsU), sort(NsU,Ns).

init([],[]).

init([fr(S,F)|Fs],[n(F,S)|Ns]) :- init(Fs,Ns).
```

```
make_tree([T],T).
make_tree([n(F1,X1),n(F2,X2)|Ns],T) :-
F is F1+F2,
insert(n(F,s(n(F1,X1),n(F2,X2))),Ns,NsR),
make_tree(NsR,T).
insert(N,[],[N]) :- !.
insert(n(F,X),[n(F0,Y)|Ns],[n(F,X),n(F0,Y)|Ns]) :- F < F0, !.
insert(n(F,X),[n(F0,Y)|Ns],[n(F0,Y)|Ns1]) :- F >= F0, insert(n(F,X),Ns,Ns1).
traverse_tree(T,Cs) :- traverse_tree(T,'',Cs1-[]), sort(Cs1,Cs), write(Cs).
traverse_tree(n(_,A),Code,[hc(A,Code)|Cs]-Cs) :- atom(A).
traverse_tree(n(_,s(Left,Right)),Code,Cs1-Cs3) :-
atom_concat(Code,'0',CodeLeft),
atom_concat(Code,'1',CodeRight),
traverse_tree(Left,CodeLeft,Cs1-Cs2),
traverse_tree(Right,CodeRight,Cs2-Cs3).
```

OUTPUT:



Conclusion:

In this Experiment ,I learned about prolog and implemented given problems.