Assignment: 4

1. Given the relations
father(X,Y) X is the father of Y
mother(X, Y) X is the mother of Y
female(X) X is female
male(X) X is male
Define prolog relations for the following:
a. sibling
b. sister
c. grandson
d. descendant
Provide some facts for the father, mother, male, and female predicates and then test the entire thing using Prolog.

Answer:

```
male(sid).
male(haresh).
male(durgesh).
male(mahesh).

female(jayati).
female(prafulla).
female(hiralaxmi).
female(vijya).
female(usha).

father(durgesh ,haresh).
father(durgesh ,mahesh).
father(haresh,jayati).
father(haresh,sid).
father(durgesh,vijya).

mother(hiralaxmi,haresh).
mother(hiralaxmi,mahesh).
mother(hiralaxmi,vijya).
mother(prafulla,sid).
mother(prafulla,jayati).


sibling(X,Y):-father(A,X),father(A,Y).
sister(X,Y):-female(X),female(Y), father(A,Y) ,father(A,X), mother(B,X),mother(B,Y).
grandson(X,Y):-male(X),father(A,X),father(Y,A).
descendant(X,Y):-father(Y,X).
```

Output:

```
?- sibling(sid,vijya).
false
?-sibling(mahesh,jayati).
false
?-sibling(sid,jayati).
true
?-grandson(durgesh,jayati).
false
?-grandson(sid,durgesh).
true
?-descendant(sid,durgesh).
false
?-descendant(mahesh,durgesh).
true
?-sister(prafulla,vijya).
false
?-sister(usha,vijya).
true
?-sister(usha,prafulla).
false
```

2. Write a Prolog relation *remove(E,L,R)* that is true if *R* is the list which results from removing one instance of *E* from list *L*. The relation is false if *E* isn't a member of *L*.
What are all of the answers to the following queries?
ask remove(a,[b,a,d,a],R).
ask remove(E,[b,a,d,a],R).
ask remove(E,L,[b,a,d]).
ask remove(p(X),[a,p(a),p(p(a)),p(p(p(a)))],R).

```
remove(X,[X],[]):-
write("\nNumber Is Not Found"),!.
remove(X,[X|Tail],[Tail]):-
write("\nNumber Is Found").
remove(X,[Y|Tail],[Y|Tail1]):-
remove(X,Tail,Tail1).
```

Output:

```
?-remove(a,[b, a, d, a],R).
R = [b, d, a]
?-remove(E,[b, a, d, a],R).
E = b,
R = [a, d, a]
?-remove(E,L,[b, a, d]).
L = [E, b, a, d]
?-remove(p(X),[a, p(a), p(p(a)), p(p(p(a)))], R).
```

```
R = [a, [p(p(a)), p(p(p(a)))]],
X = a
```

3. Write a Prolog relation *subsequence(L1,L2)* that is true if list *L1* contains a subset of the elements of *L2* in the same order.
How many different proofs are there for each of the following queries?
ask subsequence([a,d],[b,a,d,a]).
ask subsequence([b,a],[b,a,d,a]).
ask subsequence([X,Y],[b,a,d,a]).
ask subsequence(S,[b,a,d,a]).
Explain why there are that many.

```
subsequenc([],_):- !.
subsequenc([H|T],L):-
append(_,[H|TL],L),!,
subsequenc(T,TL).
```

<u>Output:</u>

```
?-subsequence([a, d],[b, a, d, a]).
true
?-subsequence([b, a],[b, a, d, a]).
true
?-subsequence([X,Y],[b, a, d, a]).
X = b,
Y = a
?-subsequence(S,[b, a, d, a]).
S = []
?-subsequence([v,s],[b, a, d, a]).
false
```

4. Write a Prolog relation that returns a list containing the union of the elements of two given lists.

```
union([], L, L).

union([H|L1T], L2, L3) :-
    memberchk(H, L2),!,
    union(L1T, L2, L3).

union([H|L1T], L2, [H|L3T]) :-
union(L1T, L2, L3T).
```

Output:

```
union_list([u,v], [w, x, y], U).
 [u,v, w, x, y]
union_list([a], [a, b, c], U).
 [a, b, c]
union_list([r, t], [y, u], U).
 [r, t, y, u]
```

5.Write another relation (anything you want) that does something not performed above.

```
list_sum([], 0).
list_sum([Head | Tail], TotalSum) :-
list_sum(Tail, Sum1),
TotalSum is Head + Sum1.
```

This will add all the number in list.

Output:

```
list_sum([1,2,3,4], S).
S = 10.
 list_sum([3, -3, 1, 5, 9], S).
S = 15.
```