Rollno :- 18MCA0514
Name :- Wasit Shafi Submitted to : Dr. Suriaya Date :.....

Q1 Define the relation
max (X, Y, Max),
So that Max is the greater of two numbers X & Y

Sol 1

```
max(X, Y, Max):-
    (
      X > Y ->
        Max is X
    ;
        Max is Y
    ).
```

Q2 Define the predicate
maxlist (List, Max)
So that Max is the greatest number in the list of numbers List.

Sol 2

```
maxlist ([], R, R).
maxlist ([X|Xs], Wk, R):- X > Wk, maxlist (Xs, X, R).
maxlist ([X|Xs], Wk, R):- X =< Wk, maxlist (Xs,  , R).
maxlist ([X|Xs], R):- maxlist (Xs, X, R).
```

Q3 Define the predicate
Sumlist (list, Max)
So that Max is the greatest number in the list of numbers list.

Sol 3

```
sumlist ([], 0).
Sumlist ([Head|Tail], Sum):- sumlist (Tail, Temp),
                             Sum is Temp + Head.
```

Date .....

Q4. The following relation classifies numbers in to three classes positive, zero & negative:
class(Number, Positive) :- Number > 0.
class(0, zero).
class(Number, negative) :- Number < 0.
Define this procedure in a more efficient way using cuts.

Sol

class(Number, Positive) :- Number > 0, !.
class(0, zero) :- !.
class(Number, negative) :- Number < 0, !.

Q5 Define the procedure split(Number, Positive, Negative) which splits a list of numbers into two list: Positive ones (including zero) & negative ones.
Eg split([3,-1,0,5,-2],[3,0,5],[-1,-2])
Propose two versions: one with a cut & one without

Sol5

i) Using cut

split([],[],[]).
split([HP|TL],[HP|TP],N) :- HP >= 0, !, split(TL,TP,N).
split([HN|TL],P,[HN|TN]) :- HN < 0, !, split(TL,P,TN).

ii) without cut.

split([],[],[]).
split([Head|Tail],[Head|list1],list2) :-
    Head >= 0, split(Tail, list1, list2).
split([Head|Tail], list1, [Head|list2]) :-
    Head < 0, split(Tail, list1, list2).

Date .....

Q. Consider this prolog program & answer the Query for a cut-free program & this program with cut. Give your answer by drawing proof tree.

```
p(X) :- a(X).
p(X) :- b(X), c(X), !, d(X), e(X).
p(X) :- b(X).
a(1).
b(1)
b(2)
c(1).
c(2)
d(2)
e(2)
b(3)
?- p(X)
```

Sol

Output (with cut) : X = 1, false

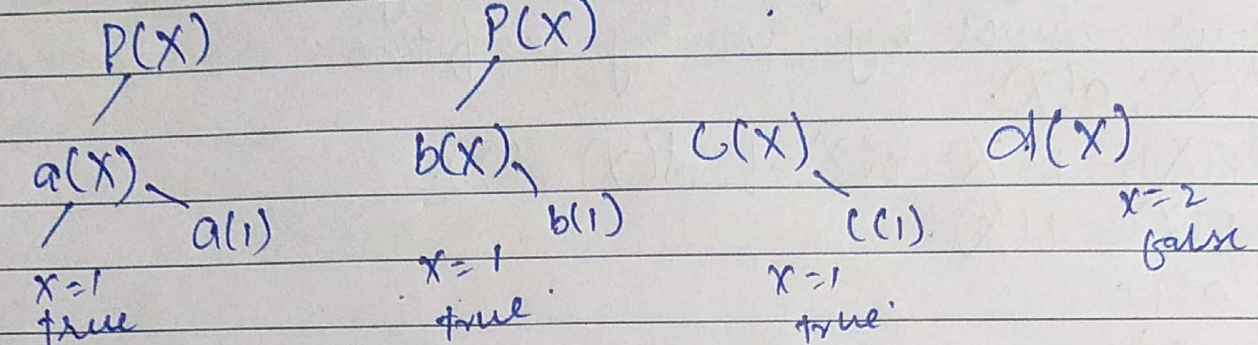
Output (without cut): X = 1
X = 2
X = 3

Cut-free code:

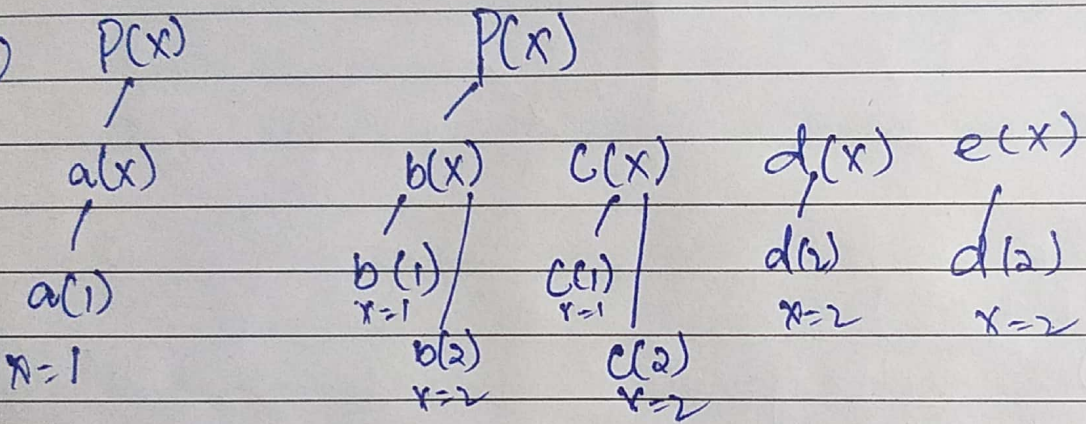
```
p(X) :- a(X).
p(X) :- b(X), c(X), d(X), e(X).
p(X) :- b(X).
a(1)
b(1).
c(1).
c(2).
d(2).
e(2).
b(3).
```

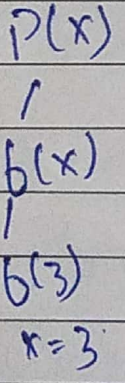
Date .....

Proof tree:



(2)





Q7 Please refer slide no 45 in lecture dated 5/5/2020 (3DCG XCLPFD to answer these Questions.

i) Whether this same is sufficient to answer Queries for both way translation? If not, modify this program to do it

ii) explain the purpose of phrase predicate in this program

iii) explain the purpose of cut operator in this program.

Date .....

i) Yes, its performing both way translations

ii) the SWI Prolog implementation of phrase/3 verifies that the list & Rest arguments are unbound, bound to the empty list or a lists cons cells. Other values raise a type error. The predicate call_dcg/3 is provided to use grammar rules with terms that are not list.

iii)

Q) Write a cryptarithmetic puzzle solver using clpfd for this puzzle.

HAIKU + SUSHI = KIMONO.

Sol)

```
:- use_module(library(clpfd)).
puzzle([H,A,I,K,U] + [S,U,S,H,I] = [K,I,M,O,N,O]):-
    Vars = [H,A,I,K,U,S,M,O,N],
    Vars ins 0..9,
    all_different(Vars),
        H*10000 + A*1000 + I*100 + K*10 + U +
        S*10000 + U*1000 + S*100 + H*10 + I #=
        K*100000 + I*10000 + M*1000 + O*100 + N*10
        + O,
        H #\= 0, S #\= 0, K #\= 0.
```