1) Enrich the KB discussed in topic A with ‘brother’ and ‘sister’ rules. [Use *not (X=Y)* to avoid reflexivity and 1-place predicate ‘male’ where necessary.]

Ans:

```
male('Borhan').

male('Ashiq').

father('Rahim','Oishe').

father('Rahim','Ashiq').

father('Rahim','Borhan').

father('Shovon','Marium').

father('Shovon','Urmi').


brother(X,Y):- father(Z,X),father(Z,Y),male(X),not(X=Y).

sister(X,Y):- father(Z,X),father(Z,Y),not(male(X)),not(X=Y).
```

2) Define a recursive procedure in English and in ProLog to find the sum of 1st n odd numbers.

Ans:

In English:

1. The sum of the 1st element of series is 1.
2. The sum of the 1st n elements of series is equal to the sum of (n-1) elements and the n th element.

In Prolog:

```
sum(1,1).

sum(N,S):- N>1,N1 is N-1,sum(N1,S1),S is S1+2*N-1.
```

3) Define a recursive procedure to find the sum of the 1st n terms of the following series.
100+105+110+…

Ans:

```
 sum(0,0).
sum(1,100).
sum(N,S) :- N1 is N-1,sum(N1,S1),S is S1+100+5*(N-1).
```

4) Define a recursive procedure to find the sum of the 1st n terms of the following series.
10x11 + 11x13 +12x15 + …

Ans:

```
sum(0,0).
sum(1,S):- S is 10*11.
sum(N,S):- N1 is N-1 , sum(N1,S1) , S is  (S1+((10+N-1)*(10+2*N-1))).
```

5) Define a recursive procedure to find the sum of 1st n terms of an equal-interval series given the 1st term and the interval. [Use a 4-place predicate.]

Ans:

```
sum(0,X,Y,S):- S is X*0+Y*0.
sum(N,F1,D,S):- N>0,N1 is N-1,sum(N1,F1,D,S1),S is S1+F1+(D*(N-1)).
```

6) Define a recursive procedure to find the length of a path between two vertices of a directed weighted graph.

Ans:

```
neighbor(i,a,35). neighbor(i,b,45). neighbor(a,c,20).
neighbor(a,d,30). neighbor(b,d,25). neighbor(b,e,35).
neighbor(b,f,27). neighbor(c,d,30). neighbor(c,g,47).
neighbor(d,g,30). neighbor(e,g,25).

pathlength(X,Y,L):-  neighbor(X,Y,L).
pathlength(X,Y,L):-  neighbor(X,Z,L1),pathlength(Z,Y,L2),L is L1+L2.
```

7) Analyze the output of the program given in section II after putting a 'cut' (!) at different places of the first 'go' clause. Also modify the code such that it displays names of people from

i) only a given town (ask to enter the name of town)

ans :

```
 lives('Karim', 'Dhaka').

lives('Ratan', 'Sylhet').

lives('Habib', 'Dhaka').

lives('Rahim', 'Khulna').

lives('Modhu', 'Bogra').

        go:- nl,!,

                write('Enter the town name: '), tab(15), read(Town),nl,nl,

                lives(Student, Town),

                write(Student),tab(15), write(Town),nl, fail.

        go.
```

ii) towns other than a given town(ask to enter the name of town).

Ans :

```
lives('Karim', 'Dhaka').
lives('Ratan', 'Sylhet').
lives('Habib', 'Dhaka').
lives('Rahim', 'Khulna').
lives('Modhu', 'Bogra').
    go1:- nl,!,
        write('Enter the town name: '), tab(15), read(Town),nl,nl,
        lives(Student, T),not(T=Town),
        write(Student),tab(15), write(T),nl, fail.
    go1.
```

8) Write a program to repeat for a given number of times the process of finding the sum of first N elements of an equal interval series given N, the interval and the first element. Ensure that all inputs are validated.

Ans :

```
go_loop1:-  write('\nHow many computations?'),read(N), doloop1(N).
    doloop1(0):-!.
    doloop1(N):- N>0, findsum, N1 is N-1, doloop1(N1).


    findsum:-      getposnum(X),    write('Enter the first number : '),  read(F1),
    write('Enter interval number : '),  read(D),  sum(X,F1,D,Y),
        write('\nThe sum of all  '), write(X), write(' term  is '),
        write(Y), write('.').


        getposnum(X):- rpt,
            write('\nEnter the N term :'),
            read(X), X>0,!.
```

```
sum(0,X,Y,S):- S is X*0+Y*0.

sum(N,F1,D,S):- N>0,N1 is N-1,sum(N1,F1,D,S1),S is S1+F1+(D*(N-1)).


rpt.

rpt:-rpt.
```

9) Write a procedure to:

i) add an element at the end of a list.

Ans :

```
go_add :-   write('Enter the List : '),   read(L),  nl,

write('Enter the element: '),  read(El),  append(L,[El],X),  write(X).
```

ii) delete the last n elements from a list.

Ans :

```
delel(1,[_|T],T).

delel(N,[_|T],L):- N>0,N1 is N-1,delel(N1,T,L).


go_delet:- write('Enter list : '),read(L), reverse(L,RL),

write('Enter n:'),read(N),delel(N,RL,L1),

reverse(L1,L3),write(L3).
```

iii) find the sum of the elements from two given positions.

Ans :

```
soe([],0).

soe([H|T],N):- soe(T,N1), N is N1+H.


delel(0,X,X).

delel(1,[_|T],T).

delel(N,[_|T],L):- N>0,N1 is N-1,delel(N1,T,L).




go_sum:- write('Enter a list : '),read(L),write('Enter start position: '),read(ST),

    write('Enter end position: '),read(E),length(L,LN),END is (LN-E),reverse(L,L3),

    delel(END,L3,L1),reverse(L1,L4),ST1 is ST-1,

    delel(ST1,L4,L2),


    soe(L2,S),

    write('The sum of '),write(L2),write(' is '),write(S).
```

10) in descending order of serial number.

Ans :

```
sort1([], []):-!.

  sort1(L1, L2):- L1=[H|T], sort1(T, T1), insert1(H, T1, L2).
```

```
insert1(El,[], [El]):-!.

insert1(El, L1, L2):-L1=[H|_],El=node(X,_,_,_),  H=node(Y,_,_,_), X >= Y, L2 = [El|L1], !.

insert1(El, L1, L2):-L1=[H|T], insert1(El, T, Lx), L2 = [H|Lx].
```

```
sort1([], []):-!.
sort1(L1, L2):- L1=[H|T], sort1(T, T1),
     insert1(H, T1, L2).
insert1(El,[], [El]):-!.
insert1(El, L1, L2):-L1=[H|_],not(El > H), L2 = [El|L1], !.
insert1(El, L1, L2):-L1=[H|T], insert1(El, T, Lx), L2 = [H|Lx].
```

11) Write Prolog codes to:

i) read a string and return the first and last symbol concatenated, and in uppercase.

Ans :

```
go5:-    write('String:'), read(Str),        string_length(Str,L),L2 is L-1,

         sub_string(Str,0,1,_,Sub),upper_lower(X,Sub),char_code(Y,X),

         sub_string(Str,L2,1,_,Sub1),upper_lower(X1,Sub1),char_code(Y1,X1),

         nl,write('The concat string: '),string_concat(Y,Y1,Str3), write(Str3).
```

ii) impose that the 1st character of an input sentence must be in uppercase.

Ans :

```
go5:-

    write('String:'), read(Str),

    string_length(Str,L),L2 is L-1,
    sub_string(Str,0,1,_,Sub),upper_lower(X,Sub),char_code(Y,X),

    sub_string(Str,1,L2,_,Sub1),string_concat(Y,Sub1,Str3),nl, write(Str3).
```

iii) find the number of words (substrings separated by one or more spaces) of an input sentence.

Ans:

```
go5:-  write('Enter the string:'), read(Str), atomic_list_concat(L,' ', Str),
      nl, length(L,X), write('Number of words: '),  write(X).
```

Session:

```
% A procedure to find the sum of the elements of a list
soe([],0).
soe([H|T],N):- soe(T,N1) , N is N1+H.

% A procedure to find the nth element
nthel(1,[H|_],H).
nthel(N,[_|T],El):- N1 is N-1, nthel(N1,T,El).

% A procedure to delete a given element
delel(H,[H|T],T).
delel(El,[H|T],L):- delel(El,T,L1), L = [H|L1].


/* Example with built in function 'length': */
avg(L,A):- soe(L,S), length(L,N), A is S/N.
```

```
/* Examples with built in function 'append': */
go_apnd1 :- append([a, b, c], [d, e], X), write(X).
go_apnd2 :- append([a, b, c], X,[a, b, c, d, e]), write(X).
go_apnd3 :- append(X,[d, e],[a, b, c, d, e]), write(X).
go_apnd4 :- append([a, b, c], [d, e],[a, b, c, d, e]).

/* Example with built in function 'member': */
goMemberCheck1 :- member(b,[a, b, c]).
goMemberCheck2 :- member(d,[a, b, c]).

/* Example with built in function 'reverse': */
go_reverse :- reverse([a, b, c], X), reverse(X,Y), write(X), nl, write(Y).



/* Example with built in function 'sort': */
go_sort:- write('Enter a list:'),read(Lt), sort(Lt, SL_ab), write(SL_ab).
```

<u>String :</u>

```
gos1:-        write('String 1:'), read(Str1),
      write('String 2:'), read(Str2),
      string_concat(Str1,Str2,Str3), write(Str3).

/* Diverse uses:  string_concat(abc, de, abcde).
               string_concat(abc, X, abcde).
               string_concat(X, de, abcde).*/


gos2:-        write('String 1:'), read(Str1),
      string_length(Str1,L), write(L).

/* Another use: string_length(abcd,4).*/
```

```
gos3:-	write('String 1:'), read(Str1),
	string_to_list(Str1,L), write(L).

/* Returns list of codes of characters.
Another use: string_to_list(X,[100,115,97,103]).*/


gos4:-	write('String:'), read(Str),
	write('Start:'), read(Pos),
	write('Length:'), read(L),
	sub_string(Str,Pos,L,LRS,Sub), nl, write(LRS), nl, write(Sub).

/* LRS - length of the remaining part.*/

gos5:-upper_lower(X,a), write(X),char_code(Y,X), write(Y).
```

Session 1:

```
go:- write('\nEnter the name of the patient:'),
	read(Patient), hypothesis(Patient, Disease), nl,
	write(Patient), write(' probably has '), write(Disease), write('.'),!.

go:- write('\nSorry, I think, I am not competent enough '), nl,
	write('to diagnose the disease.').

symptom(Patient, fever):-
	write('\nDoes '), write(Patient), write(' have a fever?(y/n)'),
	read(Reply), Reply = 'y'.

symptom(Patient, headache):-
	write('\nDoes '), write(Patient), write(' have a headache?(y/n)'),
	read(Reply), Reply = 'y'.

symptom(Patient, runny_nose):-
	write('\nDoes '), write(Patient), write(' have a runny_nose?(y/n)'),
	read(Reply), Reply = 'y'.

symptom(Patient, sneezing):-
	write('\nDoes '), write(Patient), write(' have a sneezing?(y/n)'),
	read(Reply), Reply = 'y'.
```

```
hypothesis(Patient, flu):-
    symptom(Patient, headache), symptom(Patient, fever),
    symptom(Patient, runny_nose).

hypothesis(Patient, common_cold):-
    symptom(Patient, sneezing),
    symptom(Patient, runny_nose).
```