KATHMANDU UNIVERSITY
DHULIKHEL KAVRE

COMP 301

Lab Sheet No. 4

SUBMITTED BY

Name:- Aarush Timalsina

Roll No:- 61

Group:- CE

Level:- III / I

SUBMITTED TO

Mr. Nabin Ghimire

Date of submission:- 23 /06/2021

Aarush Timalsina
Roll no : 61
CE (III / I)

Lab-4

Prolog Lists:

In prolog, list is a collection of terms, which is useful for grouping items together, or for dealing with large volumes of related data, etc. Lists are enclosed by square brackets, and items are separated by commas such as : [item 1, item 2, item 3]. They can contain repeated items or have no item at all such as: [] which is an empty list. List can even contain other list as item such as: [item1, [], [item2_1, item2_2], [3, [3.1, 3.2]]].

Parts of list:

List is composed of two parts: Head: first item of list and Tail: rest items. Prolog has a buit-in operator : pipe (|) to separate them. For example, [1 | [2, 3]] is equivalent to [1, 2, 3].

Operations on lists:

There are different operations that we can perform in list in prolog such as membership checking, calculation of length, concatenation, deleting items, appending items and inserting items, etc. In this lab, we deal with first three operations:

1. Membership checking:

In this operation, we can check if a member X is present in list or not. After setting up clause, we can run the query to check the presence. If returned true, we will know its available else, not.

2. Calculation of length:

Length of a list is the number of items that it contains. We can find the length of list by defining one predicate which will count items in list and instantiate to their number which we can print out to know. In a list [1, [2,3], 4, []] it length is 4.

3. Concatenation of lists:

Concatenation of two lists simply means joining two lists together to make one. For example, concatenation of lists: [1,2] and [3, 4, 5] results to list: [1,2,3,4,5].

Main program (lab4_list.pl):

- Clauses

  Queries(?-)
  Result (⇒)

- /* membership */

```
list_member(X, [X|Tail]).
list_member(X, [Head|Tail]) :- list_member(X,Tail).

    ?- list_member(a,[b,c,d,e,f]).
    ⇒ false.                                % couldn't find

    ?- list_member(c, [b,c,d,e,f]).
    ⇒ true;                                 % found c in list
      false.

    ?- list_member(X, [1,2,3]).
    ⇒ X = 1;                                % returned each item
      X = 2;
      X = 3;
      false.
```

- /*length of list*/

```
list_length([],0).
list_length([_|Tail],N):- list_length(Tail,N1),N is N1+1.

    ?- list_length([1,2,3,4,5],L).
    ⇒L = 5.                          /*Calculated length of list,
                                       assigned to L and displayed*/
    ?- list_length([],L).
    ⇒ L=0.                           % found length of empty list
```

- /* concatenation of two lists*/

```
concatenation([], L, L).
concatenation([X1|L1],L2,[X1|L3]):- concatenation(L1,L2,L3).

    ?- concatenation([],[1,2,3],L).
    ⇒ L= [1,2,3].                    % concat of list with empty

    ?- concatenation([a,b],[1,2,3],L).
    ⇒ L= [a,b,1,2,3].

    ?- concatenation([a,b], L,[a,b,c,1,2,3]).
    ⇒ L = [c,1,2,3,4].
```