**Institute of Information Technology (IIT)**

Jahangirnagar University



**Lab Report: 02**

Submitted by:

Name: Md. Shakil Hossain  
Roll No: 192340

Lab Date: 08/06/2023  
Submission Date: 12/06/2023

**Lab Report # Day 01**

1. Problem name : Comparison-1:

**Clause:**

*read(X),nl,*

*goal(X,Y),nl,*

*write('Section A country score is '),nl,*

*write(Y),nl,*

*write('enter section B country name'),nl,*

*read(P),nl,*

*goal(P,Q),nl,*

*write('Section B country score is '),nl,*

*write(Q),nl,*

*compare(Y,Q).*

*compare(Y,Q):-*

*Y>Q,nl,*

*write('Section A country is the winner');*

*Y<Q,nl,*

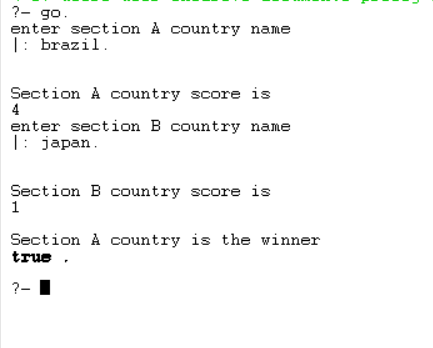
*write('Section B country is the winner');*

*Y=:=Q,nl,*

*write('Draw in both section').*

**Queries:**

**Result:**



2. Problem name : Comparison-2:

**Clause:**

*go:-*

*write('enter section A country name'),nl,*

*read(X),nl,*

*goal(X,Y),nl,*

*write('Section A country score is '),nl,*

*write(Y),nl,*

*write('enter section B country name'),nl,*

*read(P),nl,*

*goal(P,Q),nl,*

*write('Section B country score is '),nl,*

*write(Q),nl,*

*compare(Y,Q).*

*compare(Y,Q):-*

*Y>Q,nl,*

*write('Section A country is the winner');*

*Y<Q,nl,*

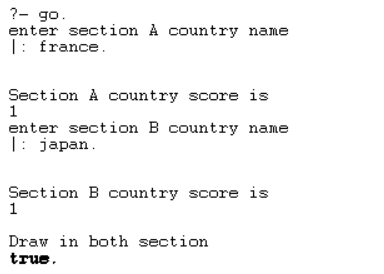
*write('Section B country is the winner');*

*Y=:=Q,nl,*

*write('Draw in both section').*

**Queries:**

**Result:**



1. Problem Name: Recursion

**Clause:**

*% Base case: Factorial of 0 is 1*

*factorial(0, 1).*

*% Recursive case: Calculate factorial of N as N multiplied by factorial of N-1*

*factorial(N, Result) :-*

*N > 0,*

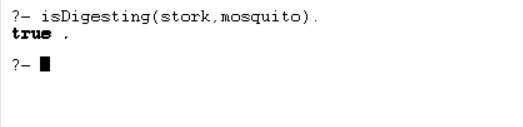
*N1 is N - 1,*

*factorial(N1, SubResult),*

*Result is N \* SubResult.*

**Queries:**

**Result:**

****

1. Problem Name: Factorial

**Clause:**

*woman(mia).*

*woman(jody).*

*woman(yolanda).*

*loves(vincent, mia).*

*loves(marsellus, mia).*

*loves(pumpkin, honey\_bunny).*

*loves(honey\_bunny, pumpkin).*

**Queries:**

?- woman(X).

?- loves(marsellus,X), woman(X).

?- loves(pumpkin,X), woman(X).

**Result:**

**A picture containing text, receipt, font, white

Description automatically generated**

1. Problem Name: Descendent

**Clause:**

*child(anna,bridget).*

*child(bridget,caroline).*

*child(caroline,donna).*

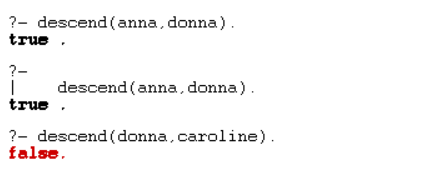
*child(donna,emily).*

*descend(X,Y):- child(X,Y).*

*descend(X,Y):- child(X,Z), descend(Z,Y).*

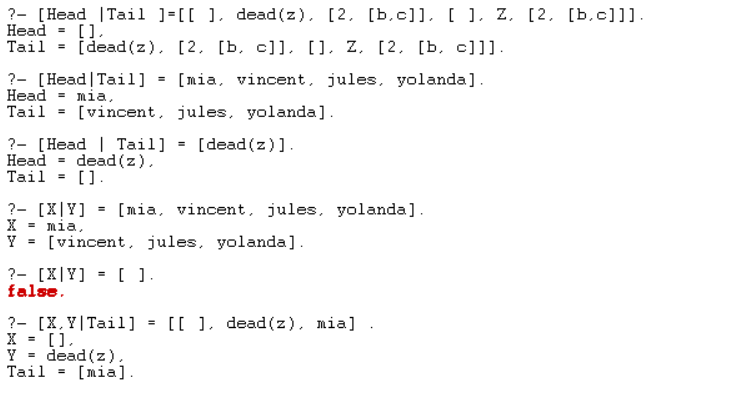
**Queries:**

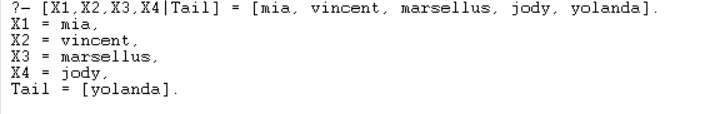
**Result:**



1. Problem Name: List

**Result:**





1. Problem Name: Member of List

**Result:**

A picture containing text, font, receipt, screenshot

Description automatically generated

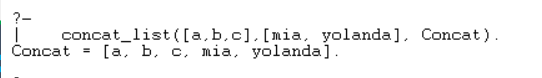
1. Problem Name: Concat List

**Clause:**

*concat\_lists(L1, L2, Concat) :- append(L1, L2, Concat)*

**Queries:**

**Result:**



1. Problem name: Delete List

**Clause:**

|  |
| --- |
| *delete\_last([\_], []). delete\_last([Head|Tail], [Head|NewTail]) :- delete\_last(Tail, NewTail).* |

**Queries:**

**Result:**



# Problem Name: Deleting from an item

**Clause:**

|  |
| --- |
| *delete\_item(X,[X|Tail],Tail).*  *delete\_item(X,[Y|Tail],[Y|Tail1]):- delete\_item(X,Tail,Tail1).* |

**Queries :**

?- delete\_item(a,[d,b,a,c],New\_list).

New\_list = [d, b, c].

**Results:**



# Problem Name: Adding an item

**Clause:**

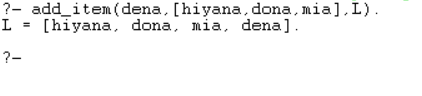
|  |
| --- |
| *add\_item(Item, List, NewList) :- append(List, [Item], NewList).* |

**Queries:**

?- add\_item(dena,[hiyana,dona,mia],L).

L = [hiyana, dona, mia, dena].

**Result:**



# Problem Name: Implement a Prolog predicate ‘equal\_length/2’ that takes two lists as input and succeeds if both lists have the same length .Give some example queries and their expected outputs.

**Clause:**

|  |
| --- |
| *equal\_length([], []).*  *equal\_length([H|T], [H1|T1]) :-*  *length([H|T]) = length([H1|T1]),*  *equal\_length(T, T1).* |

**Queries:**

?- equal\_length([], []).

?- equal\_length([1, 2, 3], [1, 2, 3]).

?- equal\_length([1, 2], [1, 2, 3]).

**Result:**

**A picture containing text, receipt, font, screenshot

Description automatically generated**

# Prolog predicate ‘maximum/3’ that takes three integers as input and returns the maximum of the three.

**Clause:**

|  |
| --- |
| *maximum(A, B, C) :-*  *A >= B,*  *A >= C,*  *write('Max = '),write(A).*  *maximum(A, B, C) :-*  *B >= A,*  *B >= C,*  *write('Max = '),write(B).*  *maximum(A, B, C) :-*  *C >= A,*  *C >= B,*  *write('Max = '),write(C).* |

**Queries:**

?- maximum(4,6,9).

?- maximum(7,200,101).

?- maximum(501,200,101).

**Result:**

A screenshot of a computer

Description automatically generated with low confidence

# 14.     Write a Prolog predicate to find the length of a list

**Clause:**

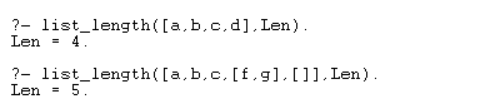
|  |
| --- |
| *list\_length([],0).*  *list\_length([\_|TAIL],N) :- list\_length(TAIL,N1), N is N1 + 1.* |

**Queries:**

?- list\_length([a,b,c,d],Len)

?- list\_length([a,b,c,[f,g],[]],Len).

**Result:**

****