

Lab: 1

Title: **Familiarization with logic programming(SWI-Prolog)**

Objectives:

1. a

Background Theory:

Procedure:

1. Installation:

- a. Link [SWI-Prolog downloads](#) was opened
- b. Suitable version was selected and downloaded
- c. The downloaded “.exe” file was installed by following the installation guide
- d. The installed file was opened

Output:

Conclusion:

Lab: 2

Title: **Consulting a knowledge in Prolog**

Objectives:

1. Implement the given family tree in knowledgebase (family.pl)
2. Consult the knowledge base in Prolog
3. Test with some queries

4. Add some rules in KB, re-consult and repeat from 3

Background Theory:

Propositional Logic

Forward and backward chaining

Production System

Knowledgebase= fact + rule

Procedure:

1. Facts and rules were added to the [knowledge base](#)

```
male(dasarath).
male(ram).
male(laxman).
male(luv).
male(kush).
male(chandraketu).
male(angada).
male(david).

female(sita).

father(dasarath,ram).
father(dasarath,laxman).
father(ram,luv).
father(ram,kush).
father(laxman,chandraketu).
father(laxman,angada).
father(luv,david).

mother(sita,luv).
mother(sita,kush).

grandfather(X,Z):-father(X,Y),father(Y,Z).
brother(X,Y):-father(Z,X),father(Z,Y),X\=Y.
uncle(X,Y):-father(Z,Y),brother(Z,X).
```

2. Knowledge base "family.pl" was consulted in SWI-Prolog

```
% c:/Users/HP/Desktop/family.pl compiled 0.00 sec, 0 clauses
```

3. Some queries were tested

```
?- male(Who).  
?- uncle(Who,david).  
?- grandfather(Who,david).  
Who = ram ,
```

Output:

```
?- male(Who).  
Who = dasarath ;  
Who = ram ;  
Who = laxman ;  
Who = luv ;  
Who = kush ;  
Who = chandraketu ;  
Who = angada ;  
Who = david.  
  
?- uncle(Who,david).  
Who = kush.  
  
?- grandfather(Who,luv).  
Who = dasarath ,  
  
?- grandfather(Who,david).  
Who = ram ,  
  
?- brother(Who,laxman).  
Who = ram ;
```

Conclusion:

Simple family tree was implemented in the knowledge base and some queries were tested to check relationship between the family members

Lab: 3

Title:**Solve the Tower of Hanoi problem in Prolog**

Objectives:

1. Implement the tower of hanoi in the knowledge base

2. Consult the knowledge base in prolog
3. Query the knowledge base for different number of disks

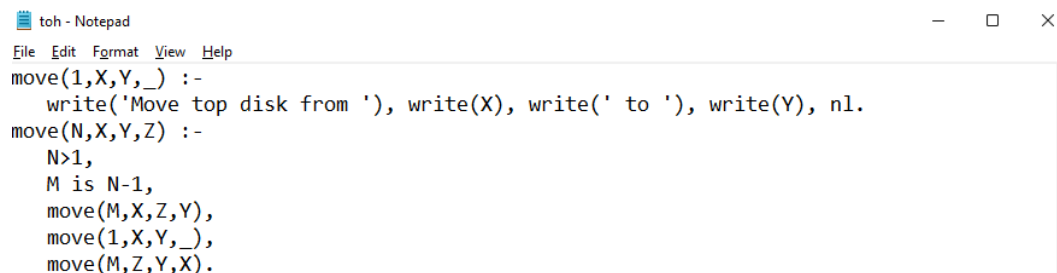
Background Theory:

Towers of Hanoi Problem is a famous puzzle where N disks are moved from a source tower to a target tower using an intermediate tower as the auxiliary holding tower. Two conditions must be followed while solving this puzzle :-

- Larger disk cannot be placed on smaller disk
- Only one disk can be moved at a time.

Procedure:

1. Program to implement the tower of hanoi problem was implemented in the [knowledge base](#)



```
File Edit Format View Help
move(1,X,Y,_):-
    write('Move top disk from '), write(X), write(' to '), write(Y), nl.
move(N,X,Y,Z):-
    N>1,
    M is N-1,
    move(M,X,Z,Y),
    move(1,X,Y,_),
    move(M,Z,Y,X).
```

2. The knowledge base was consulted in SWI-Prolog
 3. Queries were written for different number of disks
- ```
?- move(4,source,target,auxiliary).
?- move(3,source,target,auxiliary).
```

## Output:

The query was tested for tower of hanoi problems with 4 disk followed by 3 disk and the following output was received:

```

% c:/Users/HP/Desktop/toh.pl compiled 0.00 sec, 2 clauses
?- move(4,source,target,auxiliary).
Move top disk from source to auxiliary
Move top disk from source to target
Move top disk from auxiliary to target
Move top disk from source to auxiliary
Move top disk from target to source
Move top disk from target to auxiliary
Move top disk from source to auxiliary
Move top disk from source to target
Move top disk from auxiliary to target
Move top disk from auxiliary to source
Move top disk from target to source
Move top disk from auxiliary to target
Move top disk from source to auxiliary
Move top disk from source to target
Move top disk from auxiliary to target
true
Unknown action: ☐ (h for help)
Action? .

?- move(3,source,target,auxiliary).
Move top disk from source to target
Move top disk from source to auxiliary
Move top disk from target to auxiliary
Move top disk from source to target
Move top disk from auxiliary to source
Move top disk from auxiliary to target
Move top disk from source to target
true

```

It was found that the minimum number of steps required to solve the tower of hanoi problem is:  
 $2^n - 1$

## Conclusion:

Tower of Hanoi problem was solved recursively in Prolog

## Lab: X

Title: **Title**

Objectives:

4. text

Background Theory:

Output:

Procedure:

Output:

Conclusion:

