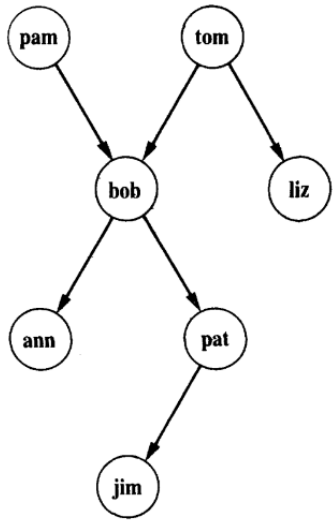
Name	**Hatim Yusuf Sawai**
UID no.	**2021300108**
Experiment No.	**5**

AIM:	**SWISH Prolog**
PROBLEM DEFINITION:	**To design and implement an intelligent system, incorporating the matching algorithm and the rule language. (family tree & knowledge generation)** **1. It should provide a fact base updating function.** **2. It should provide a function that checks the rules' LHS and return which rules were matched.** **3. It should support firing RHS according to matches.**
THEORY:	**What is Prolog?** Prolog is a programming language for symbolic, non-numeric computation. It is specially well suited for solving problems that involve objects and relations between objects. Below is an example: a family relation. The fact that Tom is a parent of Bob can be written in Prolog as: **parent(tom, bob).** Here we choose parent as the name of a relation; tom and bob are its argument. Example Family Tree: 

Syntax and Basic Fields :

In prolog, We declare some facts. These facts constitute the Knowledge Base of the system. We can query against the Knowledge Base. We get output as affirmative if our query is already in the knowledge Base or it is implied by Knowledge Base, otherwise we get output as negative. So, Knowledge Base can be considered similar to database, against which we can query. Prolog facts are expressed in definite pattern. Facts contain entities and their relation. Entities are written within the parenthesis separated by comma (,). Their relation is expressed at the start and outside the parenthesis. Every fact/rule ends with a dot (.).

Advantages :

1. Easy to build database. Doesn't need a lot of programming effort.
2. Pattern matching is easy. Search is recursion based.
3. It has built in list handling. Makes it easier to play with any algorithm involving lists.

Disadvantages :

1. LISP (another logic programming language) dominates over prolog with respect to I/O features.
2. Sometimes input and output is not easy.

Applications :

Prolog is highly used in artificial intelligence(AI). Prolog is also used for pattern matching over natural language parse trees.

CODE:	% Family Tree Facts male(john). male(bob). male(jack). male(oliver). male(ali). male(james). male(simon). male(harry). female(helen). female(sophie). female(jess). female(lily). female(mary). parent(john, mary). parent(jack,jess). parent(jack,lily). parent(helen, jess). parent(helen, lily). parent(oliver,james). parent(sophie, james). parent(jess, simon). parent(ali, simon). parent(lily, harry). parent(james, harry). % Rules sister_of(X,Y) :- female(X), parent(Z,X), parent(Z, Y), X \= Y. brother_of(X,Y) :- male(X), parent(Z,X), parent(Z, Y), X \= Y. father_of(X,Y) :- male(X), parent(X,Y). mother_of(X, Y) :- female(X), parent(X, Y). grandparent_of(X, Z) :- parent(X, Y), parent(Y, Z). aunt_of(X,Y) :- parent(Z,Y), sister_of(Z,X). uncle_of(X,Y) :- parent(Z,Y), brother_of(Z,X). % Declare facts as dynamic

```
:- dynamic(parent/2).
:- dynamic(male/1).
:- dynamic(female/1).

% Fact Base Updating
add_parent(X, Y) :- assert(parent(X, Y)).
remove_parent(X, Y) :- retract(parent(X, Y)).
add_male(X) :- assert(male(X)).
remove_male(X) :- retract(male(X)).
add_female(X) :- assert(female(X)).
remove_female(X) :- retract(female(X)).
```

OUTPUT:

1. Checking predefined predicates:

```
Hitstar53 at …\AIML-Practicals\Exp5 on main (⚠✓) via
→ cd "d:\SEM_5\AIML-Practicals\Exp5\" ; if ($?) { swipl -s main.pl }
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

1 ?- consult('main.pl').
true.

2 ?- sister_of(jess,lily).
true .

3 ?- father_of(jack,Y).
Y = jess ;
Y = lily.

4 ?- grandparent_of(jack,harry).
true.

5 ?- grandparent_of(X,harry).
X = jack ;
X = helen ;
```

2. Adding new facts and checking:

```
6 ?- add_parent(yusuf,hatim).
true.

7 ?- add_male(yusuf).
true.

8 ?- father_of(X,hatim).
X = yusuf.

9 ?- remove_parent(yusuf,hatim).
true.

10 ?- father_of(yusuf,hatim).
false.

11 ?-

Action (h for help) ? exit (status 4)
Hitstar53 at …\AIML-Practicals\Exp5 on main (⚠✓) via took 5m48s
```

CONCLUSION:	In this experiment, we learned about prolog, programming language, how to install swi-prolog and execute an intelligent system that can recognize a family tree. We implement a program with some facts, then defined some relations as rules and also fact base updating function to add our own dynamic facts.