THADOMAL SHAHANI TSEC ENGINEERING COLLEGE
Om Shete
TE C32
2103163

Experiment No : 8

Aim : Implementing Family Tree using prology.

Theory :

- Implementing a family tree in Prolog involves defining facts and rules to represent relationships between family members.
- Prolog operates using a declarative paradigm, where you specify what relationships exist rather than how to find them.
- **Facts :**
  Facts represent the basic relationships in the family tree. These are the atomic pieces of information that define who is related to whom.
- For e.g. :

```
parent(bob, alice).
parent(alice, charlie).
```

These facts state that Bob is the parent of Alice, and Alice is the parent of Charlie.

- **Rules :**
  Rules define relationship based on existing facts. They allow us to infer additional relationships from the ones we have explicitly stated.
- For e.g. :

```
father(Father, child) :-
    parent(Father, child),
    male(Father).
```

- This rule states that someone is considered a father if they are a parent of the child and if they are male.
- This allows Prolog to infer fathers based on existing parent relationships and the gender of individuals.

- Queries

Queries are used to ask questions about the relationships defined in the family tree.

- For e.g.,

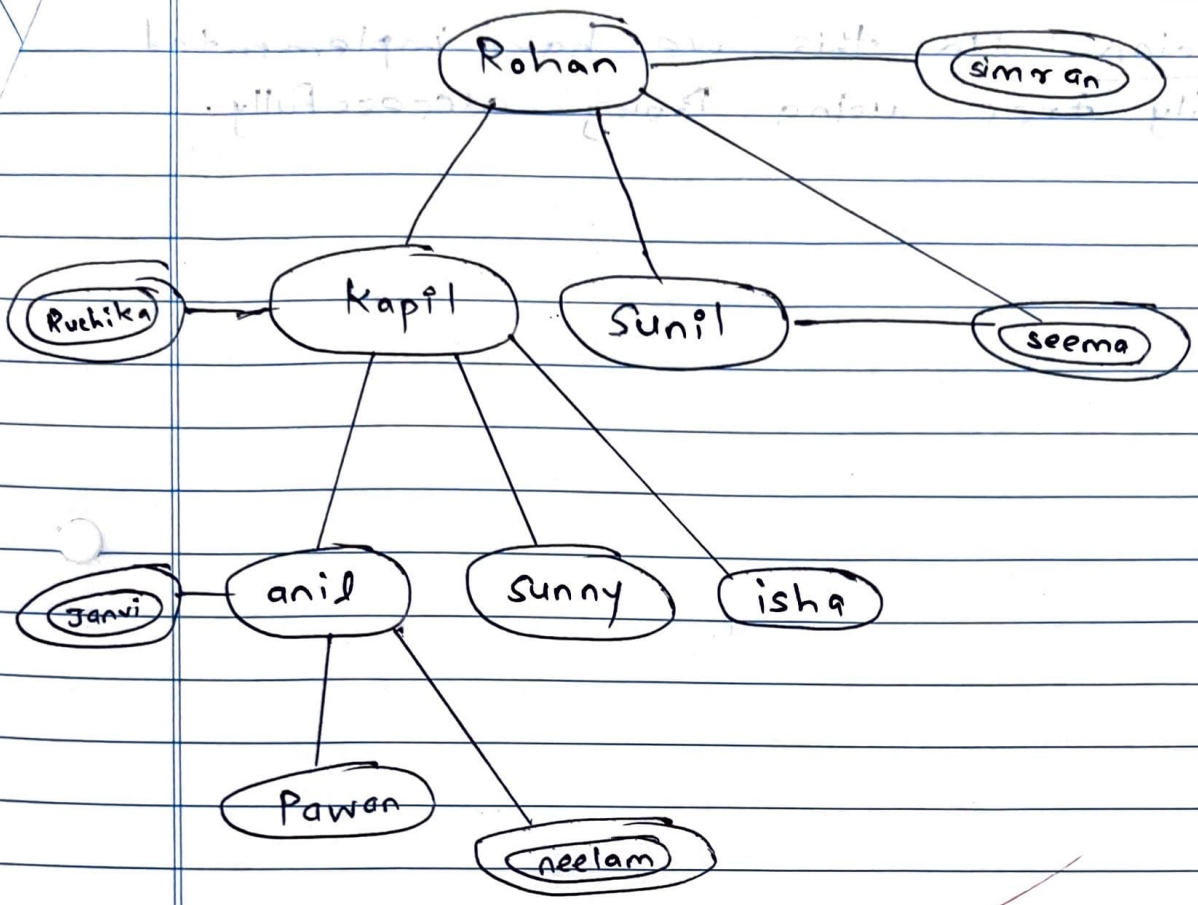
```
?- father(father, emma).
```

This query asks Prolog to find the father of Emma.

- Prolog will then search its database of facts and rules to find a matching answer.

- For e.g.,

```
?- female(X).
?- mother(X, mike).
?- mother(pat, Y).
?- father(X, Y).
```



- Dataset :

father (Rohan, Kapil)
father (Rohan, sunil)
father (Rohan, seema)
father (Kapil, anil)
father (Kapil, sunny)
father (Kapil, isha)
father (anil, neelam)
wife (simran, rohan)
wife (seema, sunil)
wife (ruchika, Kapil)
wife (janvi, anil)

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

- Conclusion : In this we have implemented Family tree using Prolog successfully.

Code:

```
% Facts: Define the relationships in the family tree
parent(john, bob).
parent(john, lisa).
parent(mary, bob).
parent(mary, lisa).
parent(bob, tom).
parent(bob, ann).
parent(lisa, patrick).
parent(lisa, emily).

% Rules: Define other relationships based on the facts
father(Father, Child) :-
    parent(Father, Child),
    male(Father).

mother(Mother, Child) :-
    parent(Mother, Child),
    female(Mother).

% Define the gender
male(john).
male(bob).
male(tom).
male(patrick).

female(mary).
female(lisa).
female(ann).
female(emily).

% Define siblings
siblings(X, Y) :-
    parent(Z, X),
    parent(Z, Y),
    X \= Y.

% Define grandparents
grandparent(Grandparent, Grandchild) :-
    parent(Grandparent, Parent),
    parent(Parent, Grandchild).
```

Output:

father(john, Child).

Child = bob
Child = lisa

```
?- father(john, Child).
```

siblings(bob, lisa).

true 1

```
?- siblings(bob, lisa).
```

grandparent(Grandparent, patrick).

Grandparent = john
Grandparent = mary

```
?- grandparent(Grandparent, patrick).
```

male(Person).

Person = john
Person = bob
Person = tom
Person = patrick

```
?- male(Person).
```

siblings(X, Y).

X = bob,
Y = lisa
X = lisa,
Y = bob
X = bob,
Y = lisa
X = lisa,
Y = bob
X = tom,
Y = ann
X = ann,
Y = tom
X = patrick,
Y = emily
X = emily,
Y = patrick

```
?- siblings(X, Y).
```