

EX.NO :6

DATE :

PROLOG

AIM :

To develop a family tree program using PROLOG with all possible facts, rules, and queries.

SOURCE CODE:

KNOWLEDGE BASE:

`/*FACTS :: */`

`male(peter).`

`male(john).`

`male(chris).`

`male(kevin).`

`female(betty).`

`female(jeny).`

`female(lisa).`

`female(helen).`

`parentOf(chris,peter).`

`parentOf(chris,betty).`

`parentOf(helen,peter).`

`parentOf(helen,betty).`

`parentOf(kevin,chris).`

`parentOf(kevin,lisa).`

`parentOf(jeny,john).`

`parentOf(jeny,helen).`

`/*RULES :: */`

`/* son,parent`

`* son,grandparent*/`

`father(X,Y):- male(Y),`

`parentOf(X,Y).`

`mother(X,Y):- female(Y),`

`parentOf(X,Y).`

```
grandfather(X,Y):- male(Y),
parentOf(X,Z),
parentOf(Z,Y).
```

```
grandmother(X,Y):- female(Y),
parentOf(X,Z),
parentOf(Z,Y).
```

```
brother(X,Y):- male(Y),
father(X,Z),
father(Y,W),
Z==W.
```

```
sister(X,Y):- female(Y),
father(X,Z),
father(Y,W),
Z==W.
```

OUTPUT:

The image displays two screenshots of a Jupyter Notebook interface, likely running Prolog queries. The top screenshot shows the notebook's file explorer and a code cell containing several Prolog queries and their results. The bottom screenshot shows the same notebook with a different set of queries and results.

Top Screenshot:

```
Male Facts:
male(peter). false

Father-Son Queries:
father(chris, peter). false
father(chris, betty). false

Grandfather Queries:
grandfather(kevin, peter). false
grandfather(jerry, peter). false

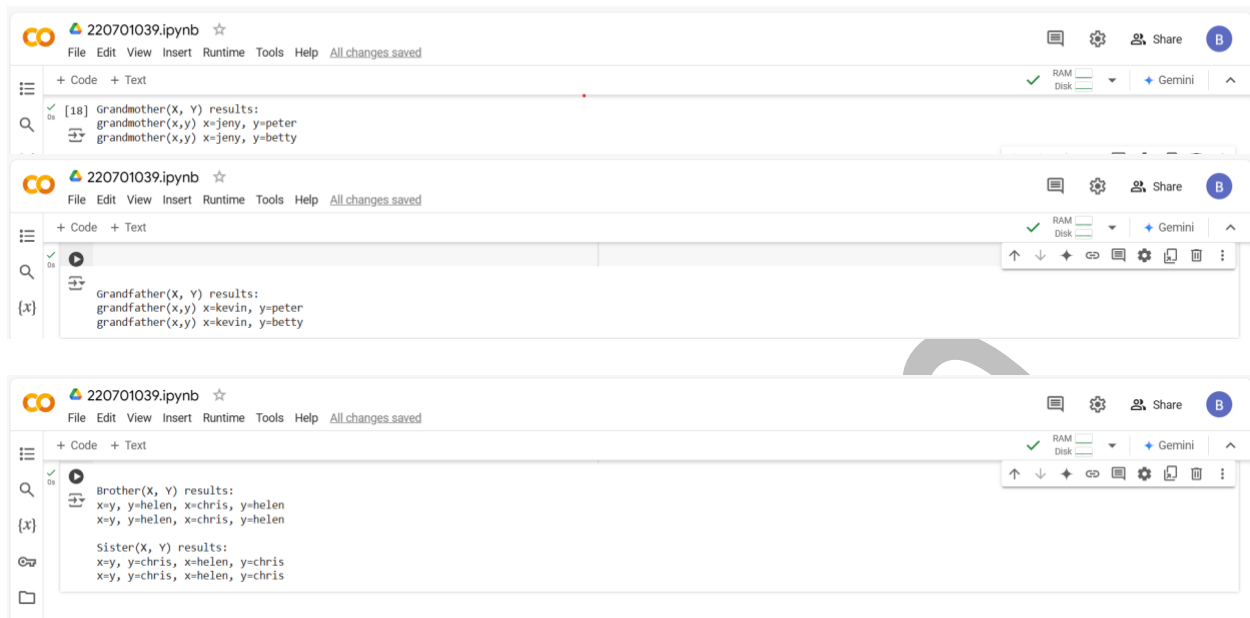
Mother-Son Queries:
mother(chris, X). false

Brother Queries:
brother(helen, chris). false
brother(chris, helen). false
```

Bottom Screenshot:

```
Father(X, Y) results:
x=chris, y=peter
x=chris, y=betty
x=kevin, y=chris
x=kevin, y=lisa

Mother(X, Y) results:
x=jenny, y=john
x=jenny, y=helen
x=helen, y=peter
x=helen, y=betty
```



RESULT :