

EX.NO: 07**DATE:****PROLOG- FAMILY TREE****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).
```

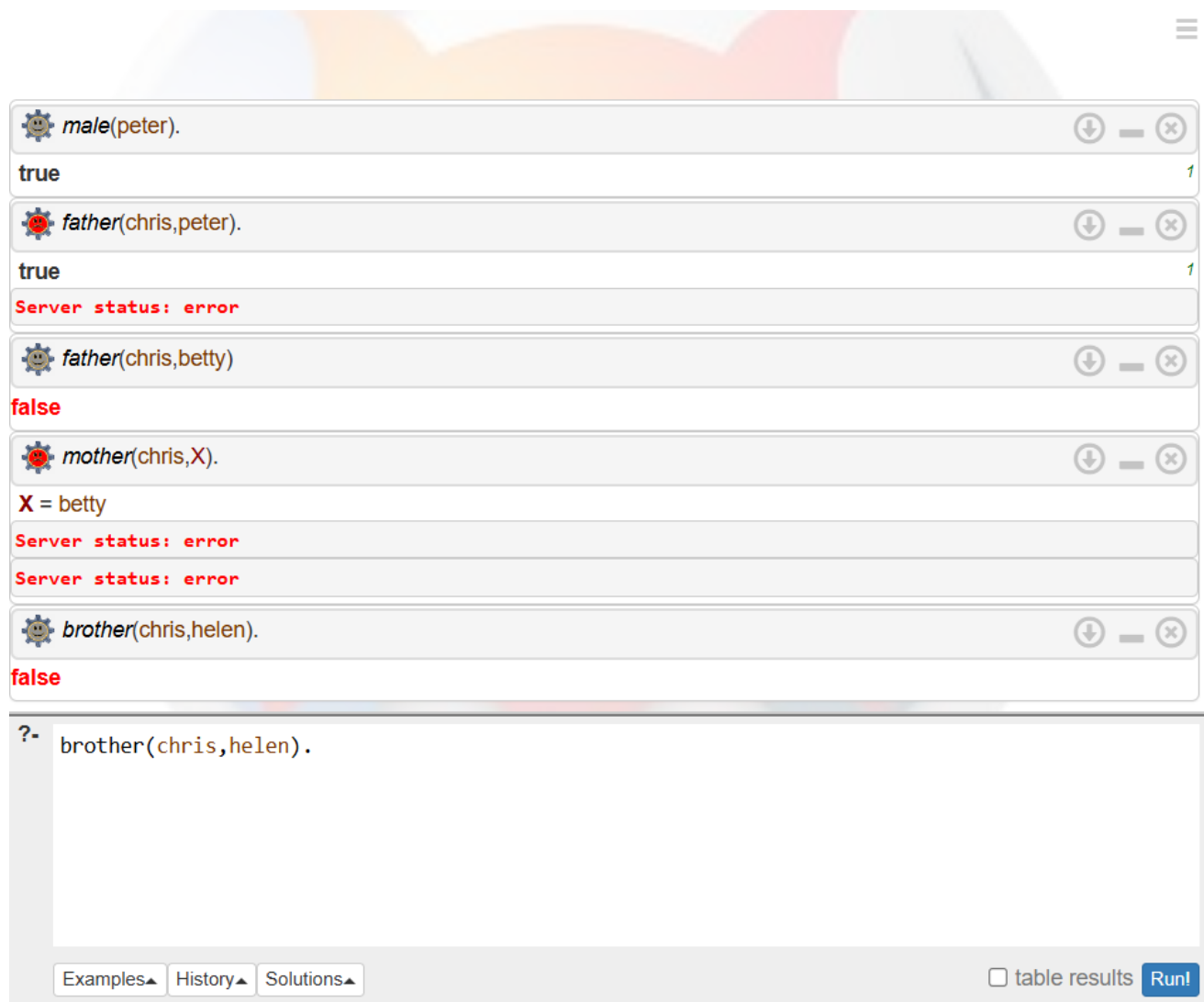
```
mother(X,Y):- female(Y), parentOf(X,Y).
```

```
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.
```



The screenshot shows a Prolog interpreter window with a header bar containing a hamburger menu icon. Below the header, there are six query panels, each with a gear icon, a query string, and control buttons (download, zoom, close). The results of the queries are displayed below each query string.

- Query: `male(peter).` Result: `true`
- Query: `father(chris,peter).` Result: `true`
- Query: `father(chris,betty)` Result: `false`
- Query: `mother(chris,X).` Result: `X = betty`
- Query: `brother(chris,helen).` Result: `false`

At the bottom of the window, there is a text input area with the query `?- brother(chris,helen).` and a button labeled `Run!`. Below the input area are three tabs: `Examples▲`, `History▲`, and `Solutions▲`. To the right of the tabs is a checkbox labeled `table results`.

RESULT:

Thus the family tree program using PROLOG with all possible facts, rules, and queries has been implemented successfully.