Write a Prolog Program for backward Chaining. Incorporate required queries.

AIM

To write a Prolog program that uses backward chaining to determine relationships such as grandparent or ancestor by querying the program, allowing Prolog to reason backward from a goal to known facts.

ALGORITHM

1. Start the program.
2. Define base facts using parent(X, Y) to represent direct relationships.
3. Define rules like grandparent(X, Z) and ancestor(X, Z) that relate facts to infer higher-level relationships.
4. Load the program into the Prolog interpreter.
5. Pose queries such as grandparent(john, tom) or ancestor(mary, tom).
6. Prolog uses backward chaining to check if the goal can be satisfied by existing facts or recursively using rules.
7. Stop.

```
% Facts
parent(john, mary).
parent(mary, susan).
parent(susan, tom).

% Rules (backward chaining: check if someone is grandparent or ancestor)
grandparent(X, Z) :- parent(X, Y), parent(Y, Z).

ancestor(X, Z) :- parent(X, Z).
ancestor(X, Z) :- parent(X, Y), ancestor(Y, Z).
```

OUTPUT:

```
?-
% c:/Users/gayathri/Downloads/parent.pl compiled 0.00 sec, 0 clauses
?-  grandparent(john, tom).
false.

?- grandparent(X, Z).
X = john,
Z = susan ;
X = mary,
Z = tom ;
false.

?- ancestor(mary, tom).
true .

?- ancestor(X, tom).
X = susan ;
X = john ;
X = mary .

?-
```

RESULT

The program successfully applies backward chaining to infer relationships. For example, it confirms that john is the grandparent of tom and identifies all ancestors of tom by reasoning backward from the goal to the known facts.