Samkit Shah

2019130060

TE Comps

Batch – C

Experiment 7

Aim: To implement the prolog problem.

Problem Statement:

Read the below passage carefully and answer the questions: Five cities all got more rain than usual this year. The five cities are: Last Stand, Mile City, New Town, Olliopolis, and Polberg. The cities are located in five different areas of the country: the mountains, the forest, the coast, the desert, and in a valley. he rainfall amounts were: 12 inches, 27 inches, 32 inches, 44 inches, and 65 inches.

1. The city in the desert got the least rain; the city in the forest got the most rain.
2. New Town is in the mountains.
3. Last Stand got more rain than Olliopolis.
4. Mile City got more rain than Polberg, but less rain than New Town.
5. Olliopolis got 44 inches of rain.
6. The city in the mountains got 32 inches of rain; the city on the coast got 27 inches of rain.

1. Which city got the most rain?
2. How much rain did Mile City get?
3. Which city is in the desert ?
4. Where is Olliopolis located?

Theory:

Prolog is a language built around the Logical Paradigm: a declarative approach to problem-solving.

There are only three basic constructs in Prolog: facts, rules, and queries.

A collection of facts and rules is called a knowledge base (or a database) and Prolog programming is all about writing knowledge bases. That is, Prolog programs simply are knowledge bases, collections of facts and rules which describe some collection of relationships that we find interesting.

So how do we use a Prolog program? By posing queries. That is, by asking questions about the information stored in the knowledge base. The computer will automatically find the answer (either True or False) to our queries.

Relationship is one of the main features that we have to properly mention in Prolog. These relationships can be expressed as facts and rules. After that we will see about the family relationships, how we can express family based relationships in Prolog, and also see the recursive relationships of the family.

Relations in Prolog

In Prolog programs, it specifies relationship between objects and properties of the objects. Suppose, there's a statement, "Amit has a bike", then we are actually declaring the ownershiprelationship between two objects — one is Amit and the other is bike.

If we ask a question, "Does Amit own a bike?", we are actually trying to find out about one relationship.

There are various kinds of relationships, of which some can be rules as well. A rule can find out about a relationship even if the relationship is not defined explicitly as a fact.

We can define a brother relationship as

follows −Two person are brothers, if,

- They both are male.
- They have the same parent.

Now consider we have the below phrases −

- parent(sudip, piyus).
- parent(sudip, raj).
- male(piyus).
- male(raj).
- brother(X,Y) :- parent(Z,X), parent(Z,Y),male(X), male(Y)

These clauses can give us the answer that piyus and raj are brothers, but we will get threepairs of output here. They are: (piyus, piyus), (piyus, raj), (raj, raj). For these pairs, given conditions are true, but for the pairs (piyus, piyus), (raj, raj), they are not actually brothers, they are the same persons. So we have to create the clauses properly to form a relationship.

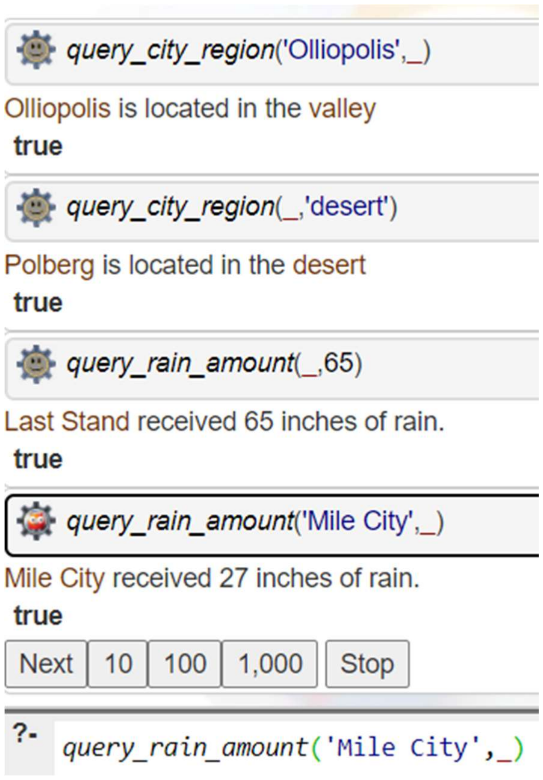
The revised relationship can be as
follows −A and B are brothers if −

- A and B, both are male
- They have same father
- They have same mother
- A and B are not same

Code:

```
program1(Pl) :-
 length(Pl, 5),
 % city names
 member(h('Last Stand', _, _), Pl),
 member(h('Mile City', _, _), Pl),
 member(h('New Town', _, _), Pl),
  member(h('Olliopolis', _, _), PI),
  member(h('Polberg', _, _), PI),
  % city areas
  member(h(_, mountains, _), PI),
  member(h(_, forest, _), PI),
  member(h(_, coast, _), PI),
  member(h(_, desert, _), PI),
  member(h(_, valley, _), PI),
  % rainfall amounts
  member(h(_, _, 12), PI),
  member(h(_, _, 27), PI),
  member(h(_, _, 32), PI),
  member(h(_, _, 44), PI),
  member(h(_, _, 65), PI),
  % Clue 1 - The city in the desert got the least rain; the city in the forest got the most rain.
  member(h(_, desert, 12), PI),
  member(h(_, forest, 65), PI),
  % Clue 2 - New Town is in the mountains.
  member(h('New Town', mountains, _), PI),
  % Clue 3 - Last Stand got more rain than Olliopolis.
  member(h('Last Stand', _, P), PI),
  member(h('Olliopolis', _, Q), PI), P > Q,
  % Clue 4 - Mile City got more rain than Polberg, but less rain than New Town.
  member(h('Mile City', _, R), PI),
  member(h('Polberg', _, S), PI), R > S,
  member(h('New Town', _, T), PI), T > R,
  % Clue 5 - Olliopolis got 44 inches of rain.
  member(h('Olliopolis', _, 44), PI),
  % Clue 6 - The city in the mountains got 32 inches of rain; the city on the coast got 27
inches of rain.
  member(h(_, mountains, 32), PI),
  member(h(_, coast, 27), PI).
  query_rain_amount(City_Name, Rainfall_Amount) :-
  program1(PI), member(h(City_Name, _, Rainfall_Amount), PI),
  write(City_Name), write(" received "), write(Rainfall_Amount),
  write(" inches of rain."), nl.
  query_city_region(City_Name, Region) :-
  program1(PI), member(h(City_Name, Region, _), PI),
  write(City_Name), write(" is located in the "), write(Region), nl.
```

Output:



```
query_city_region('Olliopolis',_)
Olliopolis is located in the valley
 true

query_city_region(_,'desert')
Polberg is located in the desert
 true

query_rain_amount(_,65)
Last Stand received 65 inches of rain.
 true

query_rain_amount('Mile City',_)
Mile City received 27 inches of rain.
 true
```

```
?- query_rain_amount('Mile City',_)
```

Conclusion: With the help of above program, I understood hoe to use prolog to solve the given problem statement and got familiar with the syntax and working of Prolog.