Experiment Number: 06

Aim: To Understand Logic Peogramming using SWI Prolog.

Lab Objective: Design and implement declarative programs in functional and logic programming languages.

Lab Outcome Mapped: Design and Develop solution based on declarative programming paradigm using functional and logic programming.

Requirements: Online/Browser based Prolog IDE provided by SWI Prolog release

Theory:

Prolog or PROgramming in LOGics is a logical and declarative programming language. It is one major example of the fourth generation language that supports the declarative programming paradigm. This is particularly suitable for programs that involve symbolic or non-numeric computation. This is the main reason to use Prolog as the programming language in Artificial Intelligence, where symbol manipulation and inference are the fundamental tasks.

In Prolog, we need not mention the way how one problem can be solved, **we just need to mention what the problem is**, so that Prolog automatically solves it. However, in Prolog we are supposed to give clues as the solution method.

Prolog language basically has three different elements −

Facts: The fact is predicate that is true, for example, if we say, “Tom is the son of Jack”, then this is a fact.

e.g. reads_a_book(saeed). // This factual clause is an assertion telling “saeed reads a book”

Rules: Rules contain conditional clauses. To satisfy a rule these conditions should be met. For example, if we define a rule as −

e.g. is_author(X) :- reads_a_book(X). // This means “if any X reads a book implies that X is author

Query: And to run a prolog program, we need some questions, and those questions can be answered by the given facts and rules.

e.g. ?- is_author(saeed). // This means “We want prolog to find out is saeed an author.”

An integral component of a prolog code is KB (Knowledge Base) it is stored in .pl file that stores collection of facts and rules about the Domain of the problem which will be solved by prolog. KB consits of Clauses that are in Horn Clausal form and can be either a fact or rule.

Performance:

Part 01:

1. Student are introduced to online swi prolog environment by accessing link.
2. Type following sample code into prolog editor.

```
likes(sam,Food) :- indian(Food), mild(Food).
likes(sam,Food) :- chinese(Food).
likes(sam,Food) :- italian(Food).
likes(sam,chips).

indian(curry).
indian(dahl).
indian(tandoori).
indian(kurma).
```

```
mild(dahl).
mild(tandoori).
mild(kurma).

chinese(chow_mein).
chinese(chop_suey).
chinese(sweet_and_sour).

italian(pizza).
italian(spaghetti).
```

3. Store this into likes.pl file.
4. Identify distinct clauses and enlist them mention parameter count in a table.
5. Query the knowledge base with following queries and enlist the output :
   a. indian(kurma). b. indian(chai). c. italian(X). d. likes(X, chips).
   e. mild(kurma). f. likes(sam,X).

Part 02:

Currently the KB has information about single person sam we need to generalise this dataset to three persons **sam, henry and john**. Also there is a dog named **gold**. Redefine the **likes** predicates to support following information: mild indian food is liked by all persons. All kind of chinese food is liked by persons. Dog likes italian food.

After adding the changes to KB store it into **likes_mod.pl** Represent 6 distinct queries and right out put for those queries.

Conclusion: We have learned how to do logic programming using SWI Prolog.

Reference:
1. Link for SWI Prolog online editor: https://swish.swi-prolog.org/example/examples.swinb
2. Prolog Tutorial: https://www.swi-prolog.org/pldoc/man?section=quickstart