**Batch: A1 Roll No.: 1911004**

**Experiment / assignment / tutorial No. 2**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **Title: Implementation of condition-action rules based agent using PROLOG** |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Objective:** Developing a basic level agent program that runs on condition-action rules

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected Outcome of Experiment:**

|  |  |
| --- | --- |
| **Course Outcome** | **After successful completion of the course students should be able to** |
| **CO1** | Understand the history & various application of AI and choose appropriate agent architecture to solve the given problem. |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. **https://www.csupomona.edu/~jrfisher/www/prolog\_tutorial/contents.html**
2. **http://www.csupomona.edu/~jrfisher/www/prolog\_tutorial/pt\_framer.html**
3. **http://www.doc.gold.ac.uk/~mas02gw/prolog\_tutorial/prologpages/**
4. **“Artificial Intelligence: a Modern Approach” by Russell and Nerving, Pearson education Publications**
5. **“Artificial Intelligence” By Rich and knight, Tata McGraw Hill Publications**
6. **“Prolog: Programming for Artificial Intelligence” by Ivan Bratko, Pearson education Publications**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Pre Lab/ Prior Concepts:** Intelligent Agent, Agent Architectures, Rule base Vs Knowledgebase approach

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Historical Profile:** Agent programs for simple applications need not be very complicated. They can be based on condition-action rules and still they give better results, though not always rational. The family tree program makes use of similar concept.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**New Concepts to be learned:**

Defining rules, using and programming with PROLOG

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

A simple agent program can be defined mathematically as an agent function which maps every possible percepts sequence to a possible action the agent can perform or to a coefficient, feedback element, function or constant that affects eventual actions:

*F*: *P* \* − >*A*

**Algorithm for ‘Condition-Action Rule Table’ Agent function:**

**function**SIMPLE-REFLEX-AGENT (percept) **returns** an action

**Static:** *rules,* a set of condition-action rules

*State*🡨INTERPRET-INPUT (percept)

*Rule*🡨*RULE-MATCH (state, rules)*

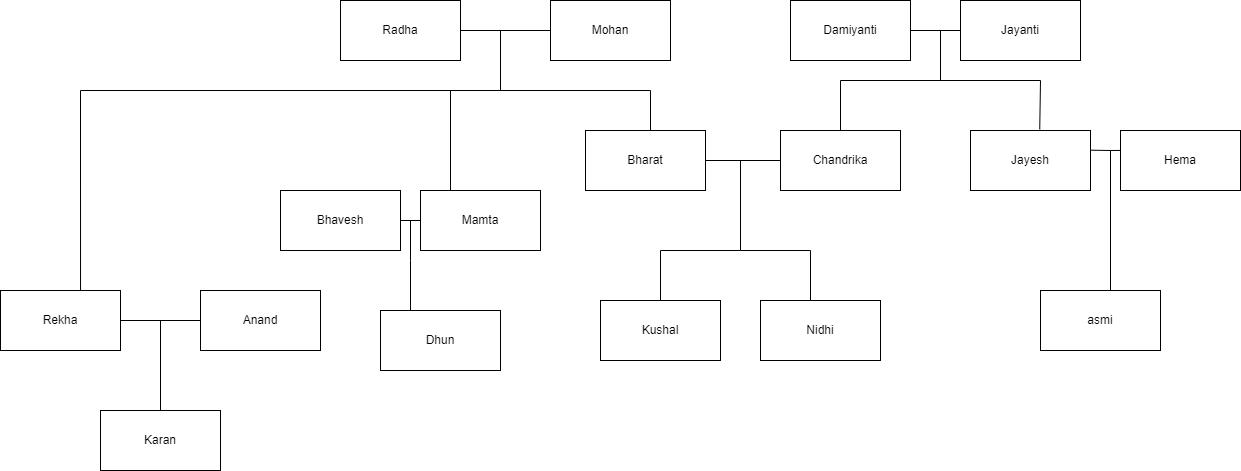
*Action*🡨*RULE-ACTION [rule]*

**Returnaction**

This approach follows a table for lookup of condition-action pairs defining all possible condition-action rules necessary to interact in an environment.

**Example Family Tree/disease-symptom mapping/ City map with their distances between them:**

**Family Tree Considered:**



**Code:**

**Base Knowledgebase:**

female(nidhi).

female(chandrika).

female(rekha).

female(radha).

female(mamta).

female(dhun).

female(damiyanti).

female(asmi).

female(hema).

male(kushal).

male(bharat).

male(mohan).

male(jayanti).

male(jayesh).

male(karan).

male(anand).

male(bhavesh).

parent(chandrika,nidhi).

parent(chandrika,kushal).

parent(bharat,kushal).

parent(bharat,nidhi).

parent(mohan,bharat).

parent(radha,bharat).

parent(jayanti,chandrika).

parent(damiyanti,chandrika).

parent(radha,mamta).

parent(radha,rekha).

parent(mohan,mamta).

parent(mohan,rekha).

parent(jayanti,jayesh).

parent(damiyanti,jayesh).

parent(jayesh,asmi).

parent(mamta,dhun).

**Rules:**

mother(X,Y):- parent(X,Y),female(X).

father(X,Y):- parent(X,Y),male(X).

sister(X,Y):- parent(Z,X),parent(Z,Y),female(X),X\==Y.

brother(X,Y):-parent(Z,X),parent(Z,Y),male(X),X\==Y.

grandmother(X,Z):- parent(X,Y),parent(Y,Z),female(X).

grandfather(X,Z):- parent(X,Y),parent(Y,Z),male(X).

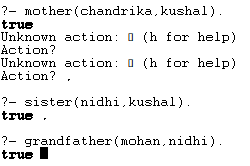
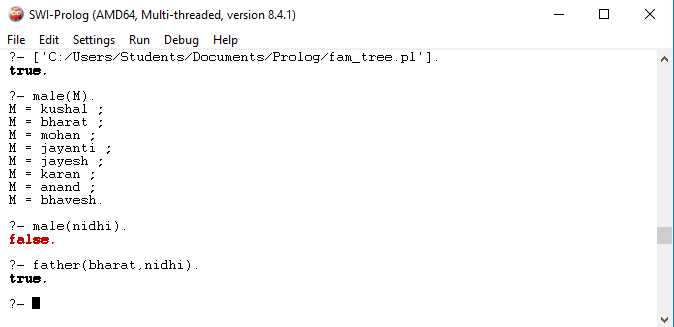
uncle(A,B):- brother(A,C),parent(C,B),male(A).

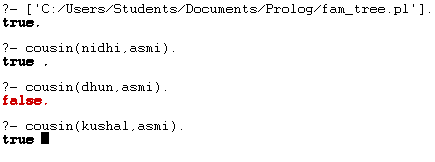
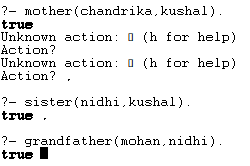
aunt(A,B):- sister(A,C),parent(C,B),female(A).

cousin(A,B):- parent(D,A),parent(C,B),mother(E,C),

mother(E,D),father(F,C),father(F,D).

**Some Sample queries and Outputs:**

****

****

**Post Lab Objective Questions**

1. **The PROLOG suit is based on**
   1. Interpreter
   2. Compiler
   3. None of the above

**Answer: a. Interpreter and b. Compiler**

1. **State true of false**

There must be at least one fact pertaining to each predicate written in the PROLOG program.

**Answer: True**

1. **State true of false**

In PROLOG program the variable declaration is a compulsory part.

**Answer: True**

**Post Lab Subjective Questions**

1. **Differentiate between a fact and a predicate with syntax.**

* A fact is a predicate expression that makes a declarative statement about the problem domain. Whenever a variable occurs in a Prolog expression, it is assumed to be universally quantified. Note that all Prolog sentences must end with a period.

Example: father(fathername, childname).

* A rule is a predicate expression that uses logical implication (:-) to describe a relationship among facts. Thus a Prolog rule takes the form left\_hand\_side :- right\_hand\_side

Eg: max(X,Y,Z) :- X>=Y -> Z=X ; Z=Y . ?- max(9, 4,Z).

Ans 🡺 Z=9

1. **Differentiate between knowledgebase and Rule base approach.**

|  |  |
| --- | --- |
| **Knowledge Base** | **Rule Base** |
| A knowledge-based system codifies human knowledge typically in a declarative format and acts upon the knowledge | A rule-based system is typically written using a rule language or as if-then statements. Some rule languages are CLIPS, OPS, ART (Automated Reasoning Tool). |
| The knowledge does not have to be written as rules. It could be written as algorithmic procedures, or cases (as in case-based reasoning), or as ontologies, or even as neural networks. The important fact is that human knowledge is codified and acted upon. | Rule based systems can also be written in Basic and C, just as long as you write a procedure that has if-then rules in it, which typically loops until some terminating condition is reached. |

1. **Differentiate between database and knowledgebase.**

|  |  |
| --- | --- |
| **Data Base** | **Knowledge Base** |
| It is a collection of data representing facts whereas a Knowledge Base contains information at a higher level of abstraction. It is building a structure to collect data. | It is a special kind of database that provides means for the computerized collection, organization and retrieval of knowledge. It is analysing many resources and inferring relations. |
| It is a structured collection of records of data or information organized in such a way that a computer program can quickly select desired pieces of data. Here, we only organize or represent those objects and their relationships through some shared information(attributes). | It is a next level in the data hierarchy, where we add some rules to manipulate these objects and their relationship for reasoning. |

1. **What is a ‘free variable’? Explain with an example.**

* The free variables in a statement stand for objects that the statement says something about. The fact that you can plug in different values for a free variable means that it is free to stand for anything.
* Eg: brother(X,Y):- parent\_of(X,A), parent\_of(Y,A),male(X) So here A is not bounded. A can be father as well as mother.