

INTRODUCTION TO PROLOG

AIM:

To learn PROLOG terminologies and write basic programs.

TERMINOLOGIES:

1. Atomic Terms:

Atomic terms are usually strings made up of lower- and uppercase letters, digits, and the underscore, starting with a lowercase letter.

Ex:

dog
ab_c_321

2. Variables:

Variables are strings of letters, digits, and the underscore, starting with a capital letter or an underscore.

Ex:

Dog
Apple_420

3. Compound Terms:

Compound terms are made up of a PROLOG atom and a number of arguments (PROLOG terms, i.e., atoms, numbers, variables, or other compound terms) enclosed in parentheses and separated by commas.

Ex:

is_bigger(elephant,X)
f(g(X,_),7)

4. Facts:

A fact is a predicate followed by a dot.

Ex:

bigger_animal(whale).
life_is_beautiful.

5. Rules:

A rule consists of a head (a predicate) and a body (a sequence of predicates separated by commas).

Ex:

is_smaller(X,Y):-is_bigger(Y,X).
aunt(Aunt,Child):-sister(Aunt,Parent),parent(Parent,Child).

SOURCE CODE:

KB1:

```
woman(mia).
woman(jody).
woman(yolanda).
playsAirGuitar(jody).
party.
Query 1: ?-woman(mia).
Query 2: ?-playsAirGuitar(mia).
Query 3: ?-party.
Query 4: ?-concert.
```

The screenshot shows a Prolog interpreter window with the following content:

```
woman(mia).
true
playsAirGuitar(mia).
false
party.
true
concert.
procedure 'concert' does not exist
```

Below the queries, there is a section for the query `?- woman(mia). playsAirGuitar(mia). party. concert.` which is currently empty. At the bottom, there are tabs for "Examples", "History", and "Solutions", and a "Run!" button.

KB2:

```
happy(yolanda).
listens2music(mia).
listens2music(yolanda):-happy(yolanda).
playsAirGuitar(mia):-listens2music(mia).
playsAirGuitar(Yolanda):-listens2music(yolanda).
```

The screenshot shows a Prolog interpreter window with the following content:

```
happy(yolanda).
true
listens2music(yolanda).
true
listens2music(mia).
true
playsAirGuitar(X).
```

Below the queries, there is a table showing the results of the query `?- playsAirGuitar(X).`:

| X | 1 |
|---------|---|
| mia | 1 |
| yolanda | 2 |

At the bottom, there are tabs for "Examples", "History", and "Solutions", and a "Run!" button.

KB3:

```

likes(dan,sally).
likes(sally,dan).
likes(john,brittney).
married(X,Y) :- likes(X,Y) , likes(Y,X).
friends(X,Y) :- likes(X,Y) ; likes(Y,X).

```

married(dan, sally).

true

likes(dan,X)

X

sally

married(john, brittney).

false

?- married(dan, sally).
likes(dan,X)
married(john, brittney).

Examples History Solutions

table results Run!

KB4:

```

food(burger).
food(sandwich).
food(pizza).
lunch(sandwich).
dinner(pizza).
meal(X):-food(X).

```

food(pizza)

true

meal(X),lunch(X)

X

sandwich

dinner(sandwich)

false

?- food(pizza)
meal(X),lunch(X)
dinner(sandwich)

Examples History Solutions

table results Run!

KB5:

```
owns(jack,car(bmw)).
owns(john,car(chevy)).
owns(olivia,car(civic)).
owns(jane,car(chevy)).
sedan(car(bmw)).
sedan(car(civic)).
truck(car(chevy)).
```

The screenshot displays a Prolog IDE interface with several query windows and a summary panel.

Query 1: `owns(John,X)`

| John | X | |
|--------|------------|---|
| jack | car(bmw) | 1 |
| john | car(chevy) | 2 |
| olivia | car(civic) | 3 |
| jane | car(chevy) | 4 |

Query 2: `owns(John,_)`

| John | |
|--------|---|
| jack | 1 |
| john | 2 |
| olivia | 3 |
| jane | 4 |

Query 3: `owns(Who,car(chevy))`

| Who | |
|------|---|
| john | 1 |
| jane | 2 |

Query 4: `owns(jane,X),sedan(X)`

false

Query 5: `owns(jane,X),truck(X)`

| X | |
|------------|---|
| car(chevy) | 1 |

Summary Panel:

?- `owns(John,X)`
`owns(John,_)`
`owns(Who,car(chevy))`
`owns(jane,X),sedan(X)`
`owns(jane,X),truck(X)`

Examples History Solutions ☒ table results Run!

RESULT:

Thus, we have written basic programs to learn prolog terminologies.