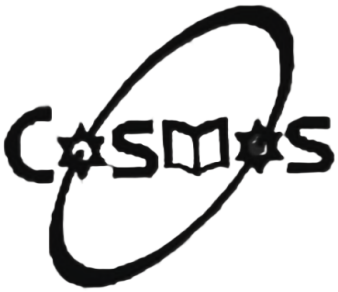
Cosmos College of Management & Technology

(Affiliated to Pokhara University)

Sitapaila, Kathmandu



Artifical Intelligence (CMP 346) Lab Report

LAB REPORT NO: 01

LAB REPORT ON:
INTRODUCTION TO PROLOG

SUBMITTED BY:-

Name: Kushal Prasad Joshi

Roll No: 230345

Faculty: Science and Technology

Group: B

SUBMITTED TO:-

Department: ICT

Lab Instructor: Mr. Ranjan Raj Aryal

Date of Experiment: 15th December 2025

Date of Submission: 21st December 2025

TITLE

INTRODUCTION TO PROLOG

OBJECTIVES

1. To understand the basic working of the SWI-Prolog environment.
2. To write and execute simple Prolog programs.

REQUIREMENTS

- SWI-prolog
- Operating System: Windows
- Text Edioter / SWI-Prolog Edioter

THEORY

Prolog is logic programming language based on formal logic. Instead of executing instructions sequentially, Prolog programs define facts and rules, and computation is performed by querying these defination.

SWI-Prolog is a widely used Prolog implementation that provides an interactive environment, compiler, and extensive libraries.

PROCEDURE

1. Install and open the SWI-Prolog environment on the computer.
2. Create a new Prolog source file with the `.pl` extension.
3. Write the required Prolog predicates (facts and rules) in the source file.
4. Save the program in an appropriate working directory.
5. Open the Prolog terminal and change the directory to the location of the program file.
6. Load the Prolog source file into the interpreter using the `consult('file_name.pl')` command.
7. Execute the required queries or predicates to observe the output.
8. Verify the output and repeat execution if necessary.

IMPLEMENTATION

```
hello:-
    format('Hello World').
```

Listing 1: Prolog Program to Display Hello World

```
?- hello.
Hello World
true.
```

Listing 2: Output of the Hello World Prolog Program

```
% ======== FACTS ==========
% schedule(day, subject)
schedule(monday,programming).
schedule(tuesday,math).
schedule(tuesday,english).
schedule(wednesday,programming).
schedule(wednesday,spanish).
schedule(thursday,circuits).
schedule(friday,none).

% difficulty(subject, level)
difficulty(programming,hard).
difficulty(math,hard).
difficulty(english,easy).
difficulty(spanish,medium).
difficulty(circuits,hard).

% ================ RULES ===========
classinformation(Day,Class,Diff) :-
    schedule(Day,Class),
    difficulty(Class,Diff).
```

Listing 3: Prolog Facts and Rules for Weekly Class Schedule and Subject Difficulty

```
?- schedule(monday,programming).
true.

?- schedule(monday, english).
false.

?- difficulty( programming, easy).
false.

?- difficulty(programming, hard).
true.

?- classinformation(tuesday, Class, Diff).
Class = math,
```

```
Diff = hard ;
Class = english,
Diff = easy.

?- classinformation(tuesday, Class, easy).
Class = english.

?- classinformation(Day, Class, hard).
Day = monday,
Class = programming ;
Day = tuesday,
Class = math ;
Day = wednesday,
Class = programming ;
Day = thursday,
Class = circuits ;
false.

?- classinformation(Day, english, hard).
false.

?- classinformation(Day, english, easy).
Day = tuesday.
```

Listing 4: Output of Queries on Class Schedule and Difficulty Information

```
% ============ FACTS =================
% Restaurant facts: cuisine(type, name)
cuisine(italian, marios).
cuisine(italian, pasta_paradise).
cuisine(mexican, taco_fiesta).
cuisine(mexican, chili_beans).
cuisine(indian, spice_route).
cuisine(japanese, sushi_zen).
cuisine(japanese, wasabi_world).
cuisine(burger, burger_kingdom).

% Price level facts: price(restaurant, level)
price(marios, expensive).
price(pasta_paradise, moderate).
price(taco_fiesta, cheap).
price(chili_beans, moderate).
price(spice_route, expensive).
price(sushi_zen, expensive).
price(wasabi_world, moderate).
price(burger_kingdom, cheap).

% Location facts: location(restaurant, area)
location(marios, downtown).
```

```
location(pasta_paradise, uptown).
location(taco_fiesta, downtown).
location(chili_beans, midtown).
location(spice_route, downtown).
location(sushi_zen, uptown).
location(wasabi_world, downtown).
location(burger_kingdom, midtown).

% Rating facts: rating(restaurant, stars)
rating(marios, 4).
rating(pasta_paradise, 3).
rating(taco_fiesta, 5).
rating(chili_beans, 4).
rating(spice_route, 5).
rating(sushi_zen, 4).
rating(wasabi_world, 3).
rating(burger_kingdom, 4).

% ================= RULES ==================
% Combined information rule
restaurantinfo(Name, Cuisine, Price, Area, Stars) :-
    cuisine(Cuisine, Name),
    price(Name, Price),
    location(Name, Area),
    rating(Name, Stars).
```

Listing 5: Prolog Facts and Rules for Restaurant Information System

```
?- cuisine(italian, marios).
true.

?- cuisine(chinese, spice_route).
false.

?- price(sushi_zen, expensive).
true.

?- price(taco_fiesta, expensive).
false.

?- location(pasta_paradise, downtown).
false.

?- cuisine(italian, Restaurant).
Restaurant = marios ;
Restaurant = pasta_paradise.

?- price(Restaurant, cheap).
Restaurant = taco_fiesta ;
```

```
Restaurant = burger_kingdom.

?- location(Restaurant, downtown).
Restaurant = marios ;
Restaurant = taco_fiesta ;
Restaurant = spice_route ;
Restaurant = wasabi_world.

?- rating(Restaurant, 5).
Restaurant = taco_fiesta ;
Restaurant = spice_route.

?-  restaurantinfo(Name, mexican, Price, Area, Stars).
Name = taco_fiesta,
Price = cheap,
Area = downtown,
Stars = 5 ;
Name = chili_beans,
Price = moderate,
Area = midtown,
Stars = 4.

?-  restaurantinfo(Name, Cuisine, expensive, downtown, Stars)
    .
Name = marios,
Cuisine = italian,
Stars = 4 ;
Name = spice_route,
Cuisine = indian,
Stars = 5 ;
false.

?-  restaurantinfo(Name, japanese, Price, Area, 4).
Name = sushi_zen,
Price = expensive,
Area = uptown ;
false.

?- restaurantinfo(taco_fiesta, Cuisine, Price, Area, Stars).
Cuisine = mexican,
Price = cheap,
Area = downtown,
Stars = 5.

?-  cuisine(Cuisine, Name), price(Name, cheap).
Cuisine = mexican,
Name = taco_fiesta ;
Cuisine = burger,
```

```
Name = burger_kingdom.

?- restaurantinfo(Name, Cuisine, moderate, downtown, Rating),
    Rating > 3.
false.

?- location(Name, uptown), rating(Name, Stars), Stars >= 4.
Name = sushi_zen,
Stars = 4.

?- restaurantinfo(Name, italian, Price, Area, Stars), (Price
   = cheap ; Price = moderate).
Name = pasta_paradise,
Price = moderate,
Area = uptown,
Stars = 3.

?-  restaurantinfo(Name, _, expensive, _, 5).
Name = spice_route ;
false.

?- restaurantinfo(Name, Cuisine, _, downtown, _), Cuisine \=
   italian.
Name = taco_fiesta,
Cuisine = mexican ;
Name = spice_route,
Cuisine = indian ;
Name = wasabi_world,
Cuisine = japanese ;
false.

?- restaurantinfo(_, mexican, cheap, _, 4).
false.
```

Listing 6: Output of Queries on Restaurant Information System

RESULT AND CONCLUSION

The Prolog programs were executed successfully in the SWI-Prolog environment, and all facts, rules, and predicates produced the expected outputs. The results confirmed correct knowledge representation and logical inference through queries. This lab helped in understanding the basics of Prolog programming, the use of facts and rules, and effective interaction with the SWI-Prolog environment, highlighting the declarative approach of logic programming.