
Statement Purpose:

This lab will introduce students to complex knowledge bases in Prolog. Students will also get in depth of prolog syntax and the process of deducing facts from knowledge base.

Activity Outcomes:

This lab teaches you the following topics:

How to use variables in Prolog
How to create complex queries
How to create complex knowledge base and use it for deduction

Instructor Note:

As pre-lab activity, read Chapter 1 from the book (Learn Prolog Now, Vol 7, by Blackburn et. al.,) to know the basics of prolog programming.

1) Stage J (Journey)

Introduction:

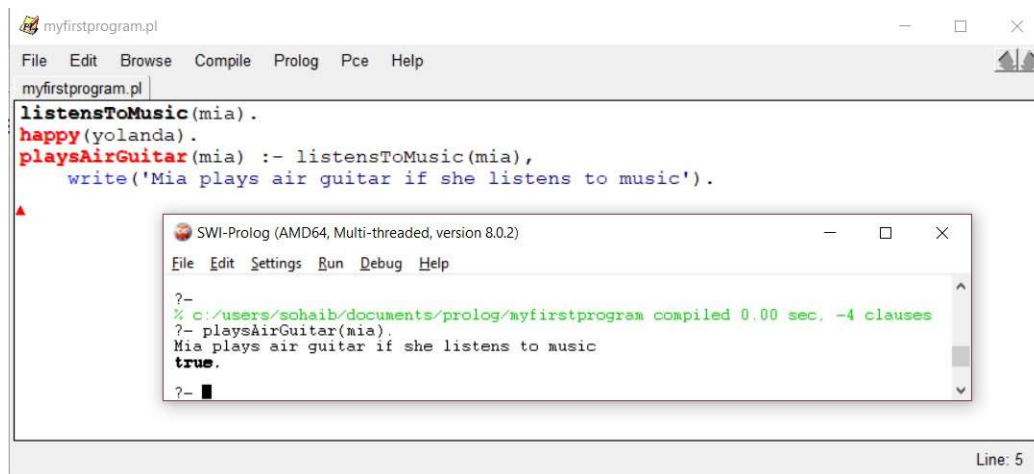
Creating complex knowledge bases poses a challenge in Prolog. It requires user to get familiar with notation and program flow which is quite different in prolog compared to other programming languages.

2) Stage a1 (apply)

Lab Activities:

Activity 1:

How to print text in prolog? Solution:



The screenshot shows the SWI-Prolog IDE with a file named `myfirstprogram.pl`. The code in the editor is:

```
listensToMusic(mia).  
happy(yolanda).  
playsAirGuitar(mia) :- listensToMusic(mia),  
    write('Mia plays air guitar if she listens to music').
```

The output window shows the following text:

```
?-  
% c:/users/sohaib/documents/prolog/myfirstprogram compiled 0.00 sec, -4 clauses  
?- playsAirGuitar(mia).  
Mia plays air guitar if she listens to music  
true.  
?-
```

The status bar at the bottom right indicates "Line: 5".

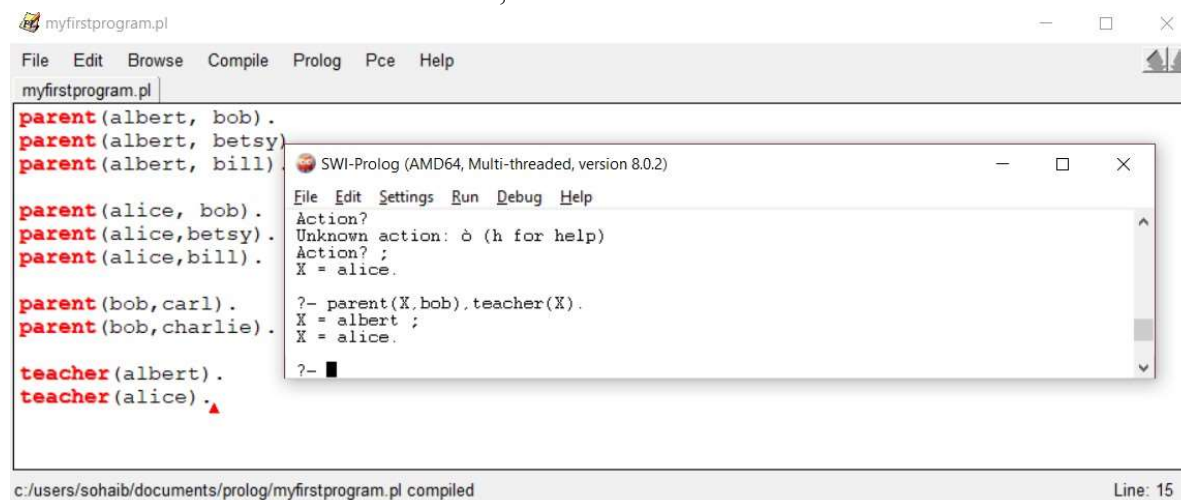
We can print custom text even within conditionals as shown above.

Activity 3:

How to ask complex queries?

Solution:

Suppose that we define a knowledgebase containing parenthood relationship alongside the teaching relationship. WE then ask the question that “give me that person X which is a parent of bob and also a teacher”. WE would do that as follows,



The screenshot shows the SWI-Prolog IDE with a file named `myfirstprogram.pl`. The code in the editor is:

```
parent(albert, bob).  
parent(albert, betsy).  
parent(albert, bill).  
  
parent(alice, bob).  
parent(alice, betsy).  
parent(alice, bill).  
  
parent(bob, carl).  
parent(bob, charlie).  
  
teacher(albert).  
teacher(alice).
```

The output window shows the following text:

```
?-  
Action?  
Unknown action: ò (h for help)  
Action? ;  
X = alice.  
  
?- parent(X, bob), teacher(X).  
X = albert ;  
X = alice.  
?-
```

The status bar at the bottom right indicates "Line: 15".

Suppose that we want to ask if Carl has a grandparent. We would do that as follows,

The screenshot shows the SWI-Prolog IDE with a file named `myfirstprogram.pl`. The main editor contains the following Prolog code:

```
parent(albert, bob).
parent(albert, betsy).
parent(albert, bill).

parent(alice, bob).
parent(alice, betsy).
parent(alice, bill).

parent(bob, carl).
parent(bob, charlie).

teacher(albert).
teacher(alice).
```

A query window is open, showing the results of the query `?- parent(X, carl), parent(Y, X).`:

```
X = albert ;
X = alice.
?- parent(X, carl), parent(Y, X).
X = bob,
Y = albert ;
X = bob,
Y = alice.
?-
```

The status bar at the bottom indicates the file is compiled and the current line is 15.

Question: Write a query in a similar fashion to determine grandchildren of Albert in above knowledgebase.

We can also define `get_GrandChild` as a rule in the knowledgebase,

The screenshot shows the SWI-Prolog IDE with the same file `myfirstprogram.pl`. The main editor now includes the `get_grandChild` rule:

```
parent(albert, bob).
parent(albert, betsy).
parent(albert, bill).

parent(alice, bob).
parent(alice, betsy).
parent(alice, bill).

parent(bob, carl).
parent(bob, charlie).

teacher(albert).
teacher(alice).

get_grandChild:-
    parent(albert, X),
    parent(X, Y),
    write('Alberts grandchild is '),
    write(Y), nl.
```

The query window shows the results of the query `?- get_grandChild.`:

```
?-
% c:/users/sohaib/documents/prolog/myfirstprogram compiled 0.00 sec. 1 clauses
?- get_grandChild.
Alberts grandchild is carl
true ;
Alberts grandchild is charlie
true ;
false.
?-
```

The status bar at the bottom indicates the rule is loaded and the current line is 15.

Let's see if Carl and Charlie share a parent,

`?- parent(X, carl), parent(X, charlie).` X = bob.

Activity 4:

We can also define variables within a consequent of a predicate which helps us to find grandparent of any X.

Solution:

The screenshot shows the SWI-Prolog IDE with a file named `myfirstprogram.pl`. The main editor contains the following Prolog code:

```
parent(albert, bob).
parent(albert, betsy).
parent(albert, bill).

parent(alice, bob).
parent(alice, betsy).
parent(alice, bill).

parent(bob, carl).
parent(bob, charlie).

get_grandParent(X, Y):-
    parent(Z, X),
    parent(Y, Z).
```

The output window shows the following execution results:

```
SWI-Prolog (AMD64, Multi-threaded, version 8.0.2)
File Edit Settings Run Debug Help
bob is the grandparent
true.
?-
% c:/users/sohaib/documents/prolog/myfirstprogram compiled 0.00 sec, -2 clauses
?- get_grandParent(carl, V).
V = albert ;
V = alice.
?-
```

The status bar at the bottom indicates: `user:get_grandParent/2: (loaded) static, 1 clause, number_of_rules(1), last_modified_generation(42286), defined` and `Line: 12`.

Activity 5:

How to use format command to print inside a Knowledge Base? Solution:

The screenshot shows the SWI-Prolog IDE with a file named `myfirstprogram.pl`. The main editor contains the following Prolog code:

```
parent(albert, bob).
parent(albert, betsy).
parent(albert, bill).

parent(alice, bob).
parent(alice, betsy).
parent(alice, bill).

parent(bob, carl).
parent(bob, charlie).

teacher(albert).
teacher(alice).

get_grandParent:-
    parent(X, carl),
    parent(X, charlie),
    format('~w ~s grandparent ~n', [X, 'is the']).
```

The output window shows the following execution results:

```
SWI-Prolog (AMD64, Multi-threaded, version 8.0.2)
File Edit Settings Run Debug Help
Warning: c:/users/sohaib/documents/prolog/myfirstprogram.pl:15:
Singleton variables: [Y]
% c:/users/sohaib/documents/prolog/myfirstprogram compiled 0.02 sec, 1 clauses
% c:/users/sohaib/documents/prolog/myfirstprogram compiled 0.00 sec, -1 clauses
?- get_grandParent.
bob is the grandparent
true.
?-
```

The status bar at the bottom indicates: `Line: 15`.

Submission requirement

Activity 6:

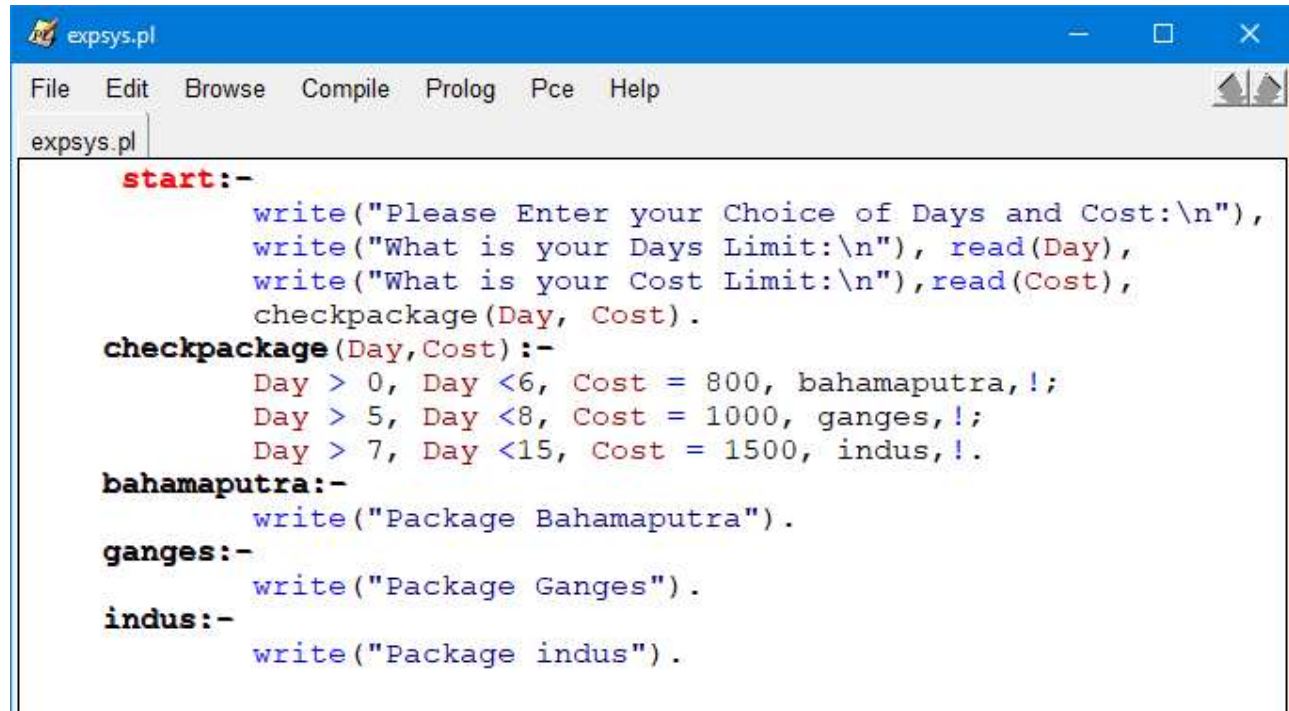
Utilizing what have learned in the previous labs, develop a working expert system for tourists based on numbers of holidays and cost:

For the coming New Year holidays a company is offering three tourist packages for visiting Pakistan, India and Bangladesh. They are called Indus, Ganges and Brahamaputra. The Indus package is for 7 days and costs \$1000, the Ganges package is for 14 days and costs \$ 1500 and the Brahamaputra package is for 5 days and costs \$800.

When run, the program should ask for the budget and the time at the disposal of the client and then suggest a package based on the time and financial considerations.

Use the predicates `write()` and `readint()`

Solution:



```
start:-
    write("Please Enter your Choice of Days and Cost:\n"),
    write("What is your Days Limit:\n"), read(Day),
    write("What is your Cost Limit:\n"), read(Cost),
    checkpackage(Day, Cost).
checkpackage(Day, Cost):-
    Day > 0, Day <6, Cost = 800, bahamaputra,!;
    Day > 5, Day <8, Cost = 1000, ganges,!;
    Day > 7, Day <15, Cost = 1500, indus,!;
    !.
bahamaputra:-
    write("Package Bahamaputra").
ganges:-
    write("Package Ganges").
indus:-
    write("Package indus").
```

The `write("<string>")` predicate displays the text on screen. Using the variables `Day` and `Cost`, the package is suggested.

Activity 7:

Utilizing what have learned in the previous labs, develop a working expert system for determining the horoscope based on date and month of birth:

Based on the date of birth and month of birth determine the zodiac sign, giving its basic information.

When run, the program should ask for the date of month and the month of the year for example if your birthday is 1st February 1900, then as Date of month you should enter "1" and as

month of the year value should be “2”. Based on this information inform the zodiac sign, that is one of the Aries, Taurus, Gemini, etc out of the 12 zodiac signs.
Use the predicates write() and readint()

Solution:



```
expsys2.pl [modified]
File Edit Browse Compile Prolog Pce Help
expsys.pl expsys2.pl [modified]

start:-
    write("What Date of the Month You were Born:\n"),read(Day),
    write("What Month of the Year You were Born:\n"),read(Month),
    check_day(Day),check_month(Month),check_sign(Day,Month).
check_day(Day):-
    Day < 0,write("date is incorrect\n");
    Day > 31,write("date is incorrect\n");
    !.
check_month(Month):-
    Month < 0,write("month is incorrect\n");
    Month > 12,write("month is incorrect\n");
    !.
check_sign(Day,Month):-
    Day =< 21, Month = 12, sagitarius.
check_sign(Day, Month):-
    Day >= 22, Month = 12, capricorn;
    Day =< 19, Month=1, capricorn.
check_sign(Day, Month):-
    Day =< 22, Month=1, acquarius;
    Day >= 19, Month=2, acquarius.

check_sign(Day, Month):-
    Day =< 22, Month=2, pisces;
    Day =< 19, Month=3, pisces.
aries:-
    write("Hello Aries\n").
sagitarius:-
    write("Hello Sagitarius\n").
capricorn:-
    write("Hello Capricorn\n").
acquarius:-
    write("Hello Acquarius\n").
pisces:-
    write("Hello pisces\n").
leo:-
```

Colourising buffer ... done, 0.02 seconds, 116 fragments

We can print custom text for each of the zodiac signs.