

Lab 9: Prolog Basics

Baheem Ferrell

Simple Fact Exercises

Exercise 1

Which of the following are syntactically correct facts (indicate yes=correct, no=incorrect)

Hazlenuts. - NO

tomsRedCar. - YES

2Ideas. - NO

Prolog. - NO

Total Score

4 out of 4 correct.

[View answers and explanation](#)

[Return to the tutorial](#)

Exercise 2

Given the database below, study the queries below it. Again indicate whether you think the goal will succeed or not by answering yes or no as prompted.

blue_box.

red_box.

green_circle.

blue_circle.

orange_triangle.

?- green_circle. - YES

?- circle_green. - NO

?- red_triangle. - NO

?- red_box. - YES

?- orange_Triangle. - NO

% Your program goes here

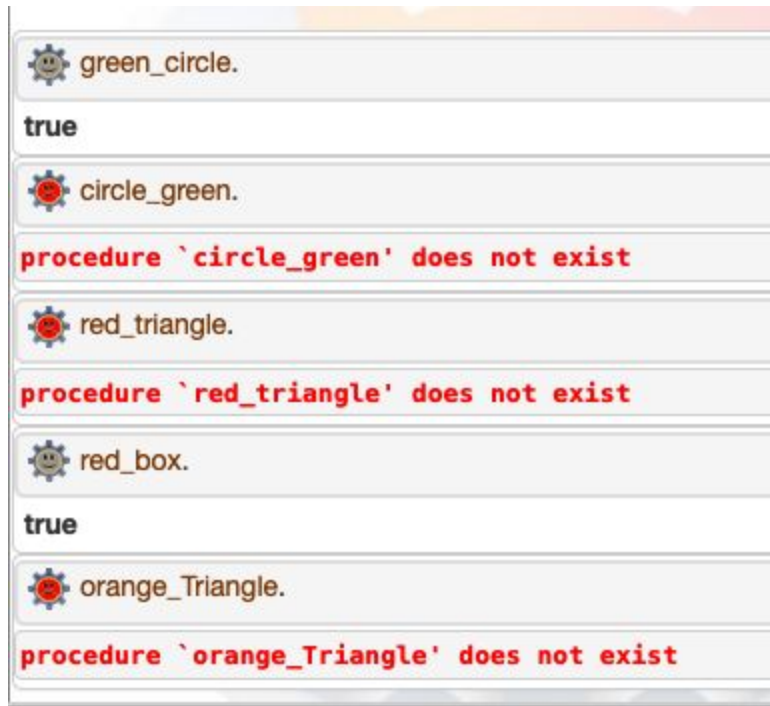
blue_box.

red_box.

green_circle.

blue_circle.

orange_triangle.



• Exercise 3 - Variable Exercise

Here are some problems for which unification sometimes succeeds and sometimes fails. The items to be compared are labeled with ¥ signs. Indicate the outcome of the proposed match *using the yes/no buttons*. If you think the unification succeeds, write down any bindings made e.g if (say) X gets bound to foo, then write X=foo as the substitution.

- eats(fred,tomatoes)
- eats(Whom,What)
YES
- eats(fred,Food)
- eats(Person,jim)
YES
- cd(29,beatles,sgt_pepper).
- cd(A,B,help).
NO
- f(X,a)
- f(a,X)
YES
- likes(jane,X)
- likes(X,jim)
NO
- f(X,Y)
- f(P,P)
YES

- f(foo,L)
 - f(A1,A1)
- YES

Exercise 4

Indicate whether the following are syntactically correct Rules.

a :- b, c, d:- e f. - NO

happy(X):- a , b. - YES

happy(X):- hasmoney(X) & has_friends(X). - NO

fun(fish):- blue(betty), bike(yamaha). - YES

Exercise 5

Given the database below, study the queries underneath it. Indicate whether you think a particular query will succeed or fail by answer yes or no *using the buttons*.

likes(john,mary).

likes(john,trains).

likes(peter,fast_cars).

likes(Person1,Person2):-

hobby(Person1,Hobby),

hobby(Person2,Hobby).

hobby(john,trainspotting).

hobby(tim,sailing).

hobby(helen,trainspotting).

hobby(simon,sailing).

?- likes(john,trains). - YES

?- likes(helen,john). - YES

?- likes(tim,helen). - NO

?- likes(john,helen). - YES

```
:- set_prolog_stack(local, limit(2 000 000)). % 1
% Your program goes here
likes(john,mary).
likes(john,trains).
likes(peter,fast_cars).

likes(Person1,Person2):-
    hobby(Person1,Hobby),
    hobby(Person2,Hobby).

hobby(john,trainspotting).
hobby(tim,sailing).
hobby(helen,trainspotting).
hobby(simon,sailing).
```



Search Exercises

Exercise 6

Consider the goal ?- fun(What)

fun(X) :- /* fun if */

 red(X), /* red */

 car(X). /* and a car */

fun(X) :- /* or its fun if its.... */

 blue(X), /* blue */

 bike(X). /* and a bike */

Below you will see pairs of goals. These may either be output of a goal e.g. red(my_hat) or calls to another goal e.g. car(my_hat).

Indicate which of the following goals you would expect to see first, for the above query, *by checking the button beside it*:

/* database of red & blue items */

red(cricket_ball).	blue(cheese).
red(my_hat).	blue(raleigh).
red(car_27).	blue(honda).

/* database of cars and bikes*/

car(peugot).	bike(yamaha).
car(rover).	bike(raleigh).
	bike(honda).

1. **fun(X) X**
 red(X)
2. blue(cheese)
 red(car_27) X
3. **bike(honda) X**
 bike(yamaha)
4. **blue(cheese) X**
 bike(raleigh)
5. bike(cheese)
 car(car_27) X

Exercise 7

Consider the following program.

```
a(X):-  
    b(X), c(X), d(X).
```

```
a(X):-  
    c(X), d(X).
```

```
a(X):-  
    d(X).
```

```
b(1).  
b(a).  
b(2).  
b(3).  
d(10).  
d(11).  
c(3).  
c(4).
```

Given the query ?- a(X). *Indicate in the box below* the successive variable bindings that the variable X gets when the above query is run, and all solutions are asked for (by using ;). *NB you should list even those bindings which do not lead to overall success, and you should list each answer even if it occurs more than once.*

Separate bindings by a comma (ie. 1,3,5.....)

1,a,2,3,3,4,10,11

Exercise 8 - Recursion Exercise

Which of the following programs uses recursion.

```
a(X):-  
b(X,Y),  
a(X).  
YES
```

```
go_home(no_12).  
go_home(X):-  
    get_next_house(X,Y),
```

home(Y).
NO

foo(X):- bar(X).
NO

lonely(X):- no_friends(X).
no_friends(X):- totally_barmy(X).
NO

has_flu(rebecca).
has_flu(john).
has_flu(X):- kisses(X,Y),has_flu(Y).
kisses(janet,john).
YES

search(end).
search(X):- path(X,Y), search(Y).
YES

Recursion Exercise 2


1. Given facts such as
Bob is taller than Mike.

Mike is taller than Jim

Jim is taller than George

Write a recursive program that will determine that Bob's height is greater than George's.

```
taller_than_rule(X,Y) :-  
    taller_than(X,Y).  
taller_than_rule(X,Y) :-  
    taller_than(X,Z),  
    taller_than_rule(Z,Y).
```

```
 taller_than_rule(bob,Who). Who = mike ; Who = jim ; Who =  
george ; false.
```

Recursion Exercises 3

town1----->-----town2----->-----town3----->-----town4----->-----town5----->-----town6

A one way road links 6 towns. Write a program that can work out if you can travel on that road.
For example. Here are two sample program behaviours.

?- can_get(town2,town5).

yes

?- can_get(town3,town1).

No

connected(town1,town2).

connected(town2,town3).

connected(town3,town4).

connected(town4,town5).

connected(town5,town6).

can_get(X,Y) :- connected(X,Y).

can_get(X,Y) :- connected(X,Z), can_get(Z,Y).

Exercise 9 - Lists Exercise

Here are some problems for which unification sometimes succeeds and sometimes fails. Decide which is the case and, if you think the unification succeeds, write down the substitutions made.

1. [a,d,z,c] and [H|T]

YES

2. [apple,pear,grape] and [A,pear|Rest]

YES

3. [a|Rest] and [a,b,c]

YES

4. [a,[]] and [A,B|Rest]

YES

5. [One] and [two|[]]

YES

6. [one] and [Two]

YES

7. [a,b,X] and [a,b,c,d]

NO