Prolog Programming Assignment #1: Various Computations

Written by **David Hennigan**

Learning Abstract

This assignment helped me become acquainted with the basics of Prolog. The assignment included 4 takes involving reproducing some demos from lessons, creating our own food facts, and coloring a map with 4 different colors where no bordering colors are the same.

Task 1 – Colors KB

Colors KB Code

```
% ------------------------------------------------------------------------
% File: colors.pro
% Line: Six color facts, structured into primaries and secondaries

% ------------------------------------------------------------------------
% primary(P) :: P is a primary color

primary(blue).
primary(red).
primary(yellow).

% ------------------------------------------------------------------------
% secondary(S) :: S is a secondary color

secondary(green).
secondary(orange).
secondary(purple).

% ------------------------------------------------------------------------
% color(C) :: C is a color

color(C) :- primary(C).
color(C) :- secondary(C).
```

Colors KB Interaction

```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).
```

```
?- primary(blue).
ERROR: Unknown procedure: primary/1 (DWIM could not correct goal)
?- consult('colors.pro').
true.

?- primary(blue).
true.

?- primary(red).
true.

?- primary(green).
false.

?- secondary(green).
true.

?- secondary(purple).
true.

?- secondary(yellow).
false.

?- color(blue).
true .

?- color(purple).
true.

?- primary(P).
P = blue ;
P = red ;
P = yellow.

?- secondary(S).
S = green ;
S = orange ;
S = purple.

?- color(C).
C = blue ;
C = red ;
C = yellow ;
C = green ;
C = orange ;
C = purple.

?- listing(primary).
```

```
primary(blue).
primary(red).
primary(yellow).

true.

?- listing(secondary).
secondary(green).
secondary(orange).
secondary(purple).

true.

?- listing(color).
color(C) :-
    primary(C).
color(C) :-
    secondary(C).

true.

?- halt.
```

Task 2 – Food KB

Food KB Code

```
% ------------------------------------------------
% File: foods.pro
% Info: This program contains 6 food facts that are
%         seperated into fruit and vegetable
% ------------------------------------------------

% ------------------------------------------------
% fruit(F) :: F is a fruit

fruit(grapefruit).
fruit(avocado).
fruit(date).

% ------------------------------------------------
% vegetable(V) :: V is a vegetable

vegetable(asperagus).
vegetable(broccoli).
vegetable(carrot).
```

```
% ---------------------------------------------------
% food(F) :: F is a food

food(F) :- fruit(F).
food(F) :- vegetable(F).
```

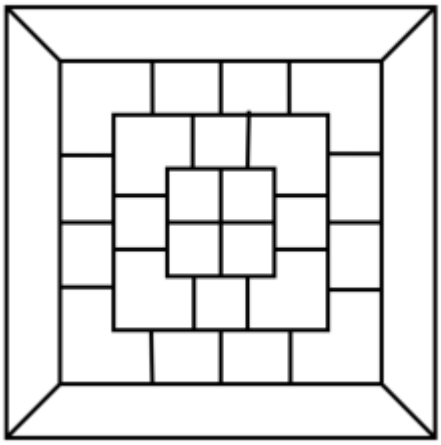
Food KB Interaction

```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- fruit(grapefruit).
ERROR: Unknown procedure: fruit/1 (DWIM could not correct goal)
?- consult('foods.pro').
true.

?- fruit(grapefruit).
true.

?- fruit(avocado).
true.

?- fruit(asperagus).
false.

?- vegetable(asperagus).
true.

?- vegetable(broccoli).
true.

?- vegetable(carrot).
true.

?- food(grapefruit).
true

?- food(carrot).
true.

?- fruit(F).
F = grapefruit ;
F = avocado ;
F = date.
```

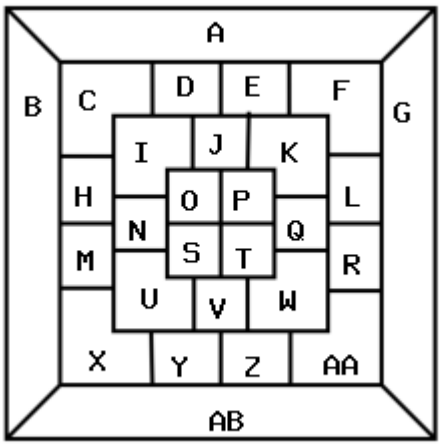
```
?- vegetable(V).
V = asperagus ;
V = broccoli ;
V = carrot.

?- food(F).
F = grapefruit ;
F = avocado ;
F = date ;
F = asperagus ;
F = broccoli ;
F = carrot.

?- listing(fruit).
fruit(grapefruit).
fruit(avocado).
fruit(date).

true.

?- listing(vegetable).
vegetable(asperagus).
vegetable(broccoli).
vegetable(carrot).

true.

?- listing(food).
food(F) :-
    fruit(F).
food(F) :-
    vegetable(F).

true.

?- halt.
```

Task 3 – Map Coloring

The Given Map



The Labeled Map



Code for Coloring the Map

```
% ----------------------------------------------
% File: mapcoloring.pro
% Info: This program is used to color a custom
%       map with 4 colors, namely, red, blue,
%       magenta, and yellow.
% ----------------------------------------------

% ----------------------------------------------
% different(X,Y) :: X is not equal to Y

different(red,blue).
different(red,magenta).
different(red,yellow).
different(blue,red).
```

```
different(blue,magenta).
different(blue,yellow).
different(magenta,red).
different(magenta,blue).
different(magenta,yellow).
different(yellow,red).
different(yellow,blue).
different(yellow,magenta).

% ----------------------------------------------
% coloring(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z,AA,AB)

coloring(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z,AA,AB) :-
    different(A,B),
    different(A,C),
    different(A,D),
    different(A,E),
    different(A,F),
    different(A,G),
    different(B,C),
    different(B,H),
    different(B,M),
    different(B,X),
    different(B,AB),
    different(C,D),
    different(C,I),
    different(C,H),
    different(D,I),
    different(D,J),
    different(D,E),
    different(E,J),
    different(E,K),
    different(E,F),
    different(F,K),
    different(F,L),
    different(F,G),
    different(G,L),
    different(G,R),
    different(G,AA),
    different(G,AB),
    different(H,I),
    different(H,M),
    different(H,N),
    different(I,N),
    different(I,O),
    different(I,J),
    different(J,O),
    different(J,P),
    different(J,K),
```

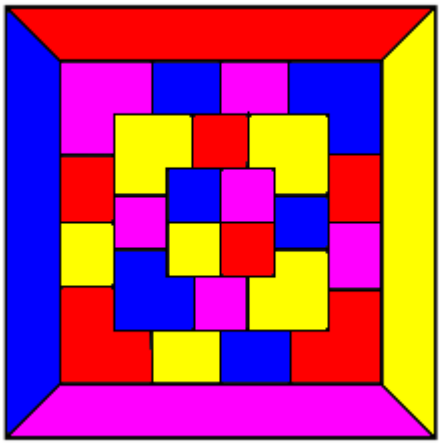
```
    different(K,P),
    different(K,Q),
    different(K,L),
    different(L,Q),
    different(L,R),
    different(M,N),
    different(M,U),
    different(M,X),
    different(N,U),
    different(N,S),
    different(N,O),
    different(O,P),
    different(O,S),
    different(P,T),
    different(P,Q),
    different(Q,T),
    different(Q,W),
    different(Q,R),
    different(R,W),
    different(R,AA),
    different(S,U),
    different(S,V),
    different(S,T),
    different(T,V),
    different(T,W),
    different(U,X),
    different(U,Y),
    different(U,V),
    different(V,Y),
    different(V,Z),
    different(V,W),
    different(W,Z),
    different(W,AA),
    different(X,AB),
    different(X,Y),
    different(Y,AB),
    different(Y,Z),
    different(Z,AB),
    different(Z,AA),
    different(AA,AB).
```

Map Coloring Interaction

```
?- consult('mapcoloring.pro').
true.

?- coloring(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z,AA,AB).
A = H, H = J, J = L, L = T, T = X, X = AA, AA = red,
```

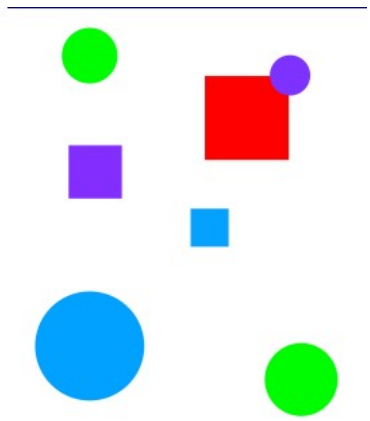
B = D, D = F, F = O, O = Q, Q = U, U = Z, Z = blue,
C = E, E = N, N = P, P = R, R = V, V = AB, AB = magenta,
G = I, I = K, K = M, M = S, S = W, W = Y, Y = yellow.

The Colored Map



Task 4 – Floating Shapes World KB

Floating Shapes World Image



Floating Shapes World KB Code

```
% ----------------------------------------------
% -- File: shapes_world_1.pro
% -- Line: Loosely represented 2-D shapes world
% ----------------------------------------------

% ----------------------------------------------
```

```
% -- Facts
% ---------------------------------------------

% ---------------------------------------------
% --- square(N,side(L),color(C)) :: N is the name
% --- of a circle with side L, and color C

square(sera,side(7),color(purple)).
square(sara,side(5),color(blue)).
square(sarah,side(11),color(red)).

% ---------------------------------------------
% --- circle(N,radius(R),color(C)) :: N is the name
% --- of a circle with radius R and color C

circle(carla,radius(4),color(green)).
circle(cora,radius(7),color(blue)).
circle(connie,radius(3),color(purple)).
circle(claire,radius(5),color(green)).

% ---------------------------------------------
% Rules
% ---------------------------------------------

% ---------------------------------------------
% --- circles :: list the names of all the circles

circles :- circle(Name,_,_), write(Name),nl,fail.
circles.

% ---------------------------------------------
% --- squares :: list the names of all of the squares

squares :- square(Name,_,_), write(Name),nl,fail.
squares.

% ---------------------------------------------
% --- shapes :: list the names of all of the shapes

shapes :- circles,squares.

% ---------------------------------------------
% --- blue(Name) :: Name is a blue shape

blue(Name) :- square(Name,_,color(blue)).
blue(Name) :- circle(Name,_,color(blue)).

% ---------------------------------------------
% --- large(Name) :: Name is a large shape
```

```
large(Name) :- area(Name,A), A >= 100.

% ---------------------------------------------
% --- small(Name) :: Name is a small shape

small(Name) :- area(Name,A), A < 100.

% ---------------------------------------------
% --- area(Name,A) :: A is the area of the shape with name Name

area(Name,A) :- circle(Name,radius(R),_), A is 3.14 * R * R.
area(Name,A) :- square(Name,side(S),_), A is S * S.
```

Floating Shapes World KB Interaction

```
?- consult('shapes_world_1.pro').

true.


?- listing(squares).

squares :-
    square(Name, _, _),
    write(Name),
    nl,
    fail.

squares.


true.


?- squares.

sera

sara

sarah

true.


?- listing(circles).
```

```
circles :-
    circle(Name, _, _),
    write(Name),
    nl,
    fail.
circles.

true.

?- circles.
carla
cora
connie
claire
true.

?- listing(shapes).
shapes :-
    circles,
    squares.

true.

?- shapes.
carla
cora
connie
claire
sera
sara
```

sarah

true.

?- blue(Shape).

Shape = sara ;

Shape = cora.

?- large(Name),write(Name),nl,fail.

cora

sarah

false.

?- small(Name),write(Name),nl,fail.

carla

connie

claire

sera

sara

false.

?- area(cora,A).

A = 153.86 .

?- area(carla,A).

A = 50.24 .

?- halt.

Annotated Demo portion

?- consult('shapes_world_1.pro').

```
true.

?- listing(squares).
squares :-
    square(Name, _, _),
    write(Name),
    nl,
    fail.
squares.

true.

?- squares.
sera
sara
sarah
true.

?- listing(circles).
circles :-
    circle(Name, _, _),
    write(Name),
    nl,
    fail.
circles.

true.

?- circles.
carla
```

```
cora
connie
claire
true.

?- listing(shapes).
shapes :-
    circles,
    squares.

true.

?- shapes.
carla
cora
connie
claire
sera
sara
sarah
true.

?- blue(Shape).
Shape = sara ;
Shape = cora.

?- large(Name),write(Name),nl,fail.
cora
sarah
false.
```

?- small(Name),write(Name),nl,fail.

carla

connie

claire

sera

sara

false.

?- area(cora,A).

A = 153.86 .

?- area(carla,A).

A = 50.24 .

?- halt.