Prolog Programming Assignment #1: Various Computations

Learning Abstract

In this assignment we learn about a very simple KB pertaining to colors in Task 1 and 2. In task 3, we paint the map coloring by using four different colors only. And in task 4, we establish and interact with a given KB of a bit more complexity than the first task.

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

```
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.
```

Task 2 - Food KB

Food KB Code

```
% ---------------------------------------------------------------------
% File: foods.pro
% Line: Six food facts, structured into fruit and vegetable
% ---------------------------------------------------------------------
% fruit(P) :: P is a fruit
fruit(grapefruit).
fruit(avocado).
fruit(date).
% ---------------------------------------------------------------------
% vegetable(S) :: S is a vegetable
vegetable(asperagus).
vegetable(broccoli).
vegetable(carrot).
% ---------------------------------------------------------------------
% food(C) :: C is a food
food(C) :- fruit(C).
food(C) :- vegetable(C).
```

Food KB Interaction

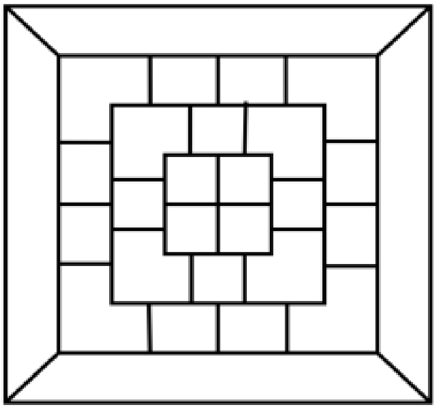
```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.1.8)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

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)
?-
% c:/Users/DELL/Documents/Prolog/foods.pro compiled 0.00 sec, 8 clauses
?- fruit(grapefruit).
true.

?- fruit(avocado).
true.

?- fruit(mango).
false.
```

```
?- vegetable(asperagus).
true.

?- vegetable(carrot).
true.

?- vegetable(tomato).
false.

?- food(date).
true .

?- food(carrot).
true.

?- fruit(P).
P = grapefruit ;
P = avocado ;
P = date.

?- vegetable(S).
S = asperagus ;
S = broccoli ;
S = carrot.

?- food(C).
C = grapefruit ;
C = avocado ;
C = date ;
C = asperagus ;
C = broccoli ;
C = carrot.
?- listing(fruit).
fruit(grapefruit).
fruit(avocado).
fruit(date).

true.

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

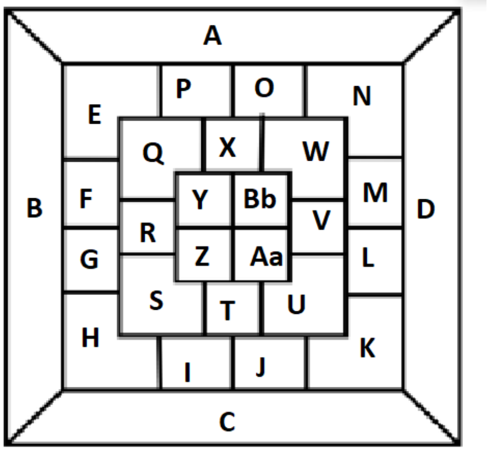
```
true.

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

True.
```

Task 3 - Map Coloring

The Given Map



The Labeled Map



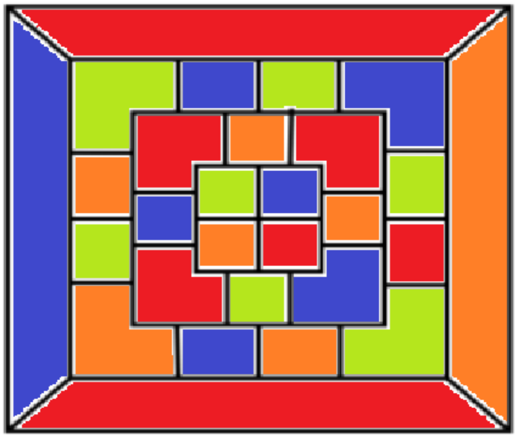
Code for Coloring the Map

```
% ---------------------------------------------------------------------------
% File: map.pro
% Line: Program to find a 4-color map rendering for a given map.
% More: The colors used will be red, blue, green, and orange.
% More: The alphabet abbreviations are used to stand for the boxes.
% ---------------------------------------------------------------------------
% different(X,Y) :: X is not equal to Y

different(red,blue).
different(red,green).
different(red,orange).
different(green,blue).
different(green,orange).
different(green,red).
different(blue,green).
different(blue,orange).
different(blue,red).
different(orange,blue).
different(orange,green).
different(orange,red).

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

Map Coloring Interaction

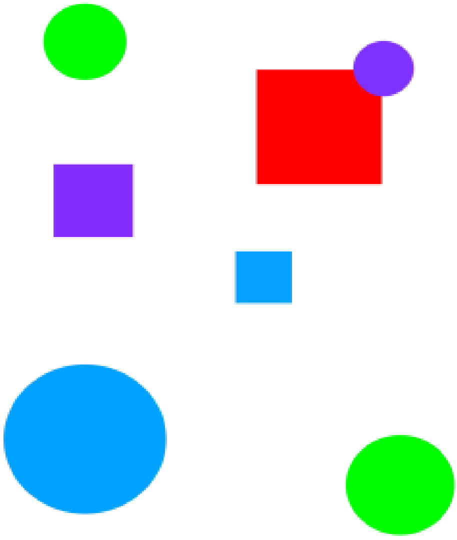
```
?-
% c:/Users/DELL/Documents/Prolog/map.pl compiled 0.00 sec, 13 clauses
?- coloring(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z,Aa,Bb).
A = C, C = L, L = Q, Q = S, S = W, W = Aa, Aa = red,
B = I, I = N, N = P, P = R, R = U, U = Bb, Bb = blue,
D = F, F = H, H = J, J = V, V = X, X = Z, Z = orange,
E = G, G = K, K = M, M = O, O = T, T = Y, Y = green
```

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 (simple take on SHRDLU) %
------------------------------------------------------------------------- %
------------------------------------------------------------------------- % --- Facts ... %
------------------------------------------------------------------------- %
------------------------------------------------------------------------- % --- square(N,side(L),color(C)) :: N is the name of a
square 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 of 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. %
------------------------------------------------------------------------- % --- squares :: 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

```
?-
% c:/Users/DELL/Documents/Prolog/shapes_world.pro compiled 0.00 sec, 0 clauses
?- 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
```