Temitope Emokpae 4/6/2023

CSC 344

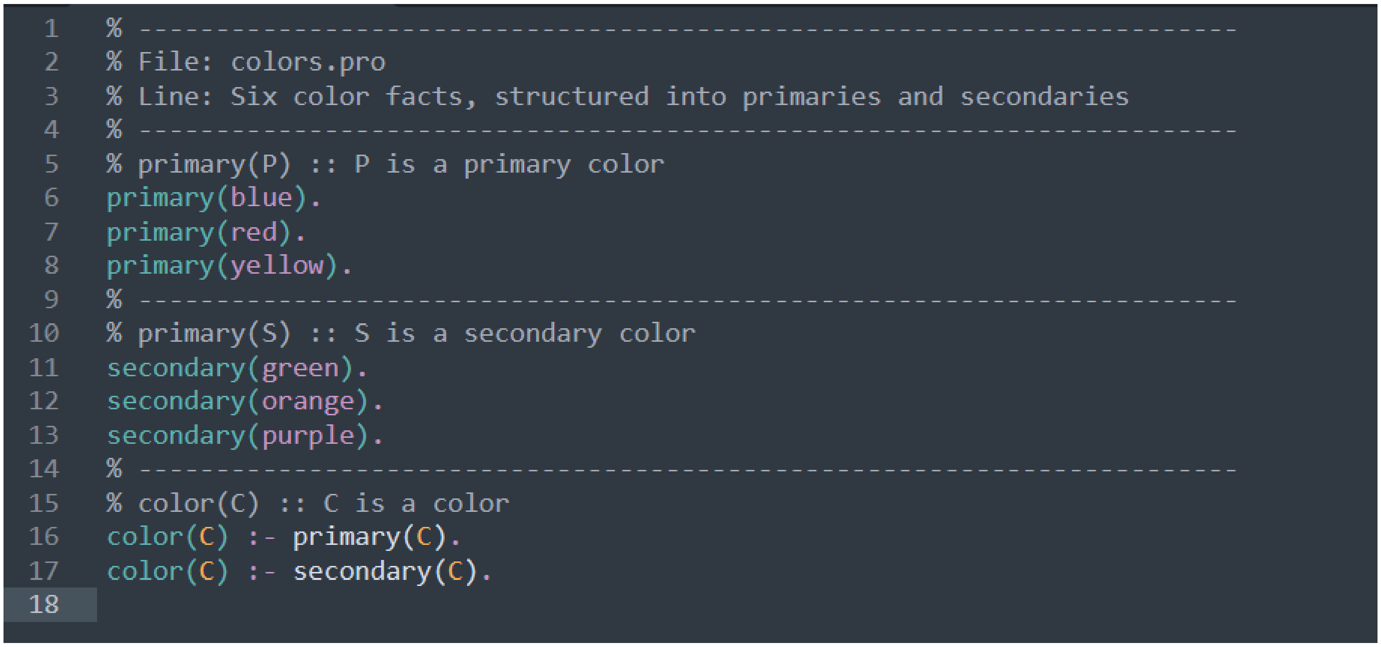
Prolog Programming Assignment #1: Various Computations

Learning Abstract

Task 1 involves establishing and interacting with the knowledge base detailed in Prolog Lesson 1, a very simple KB pertaining to colors. Task 2 involves establishing and interacting with a very simple KB which is structurally just like the given KB of Task 1, but which you are asked to piece together yourself, one pertaining to food. Task 3, based on Prolog Lesson 3, is all about solving a map coloring problem. Task 4 involves establishing and interacting with a given KB of a bit more complexity than that featured in the first task. This is the KB about floating shapes, inspired by Terry Winograd's blocks world, that was presented in Prolog Lesson 4. Collectively, these tasks afford an opportunity to get acquainted with the basics of Prolog programming.

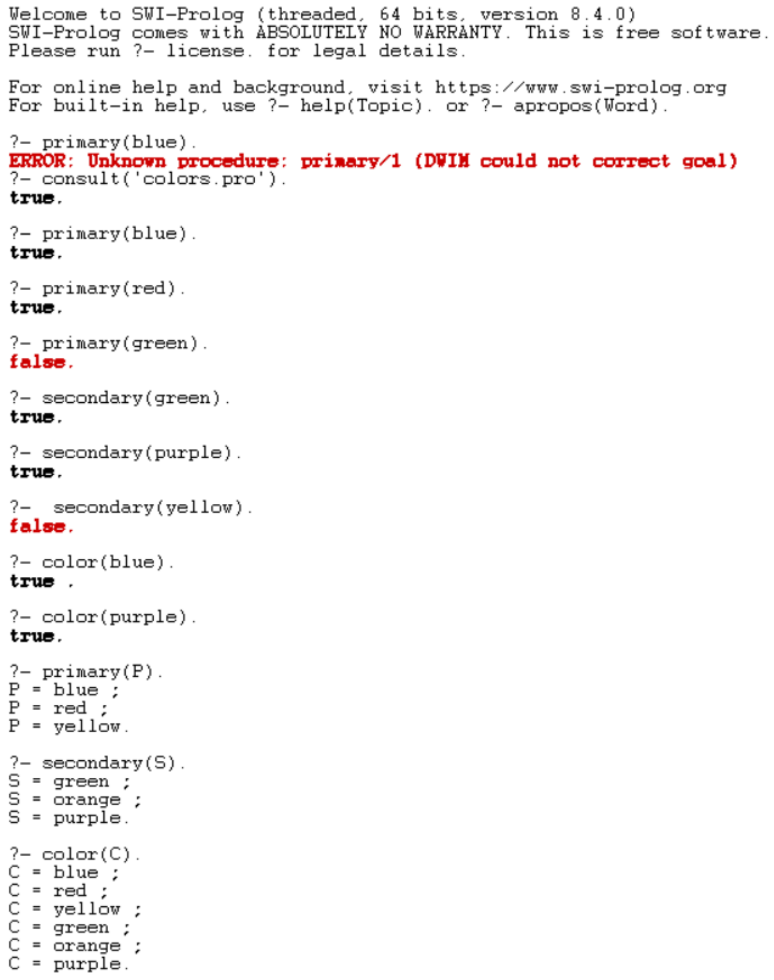
Task 1 - Colors KB

Colors KB Code:

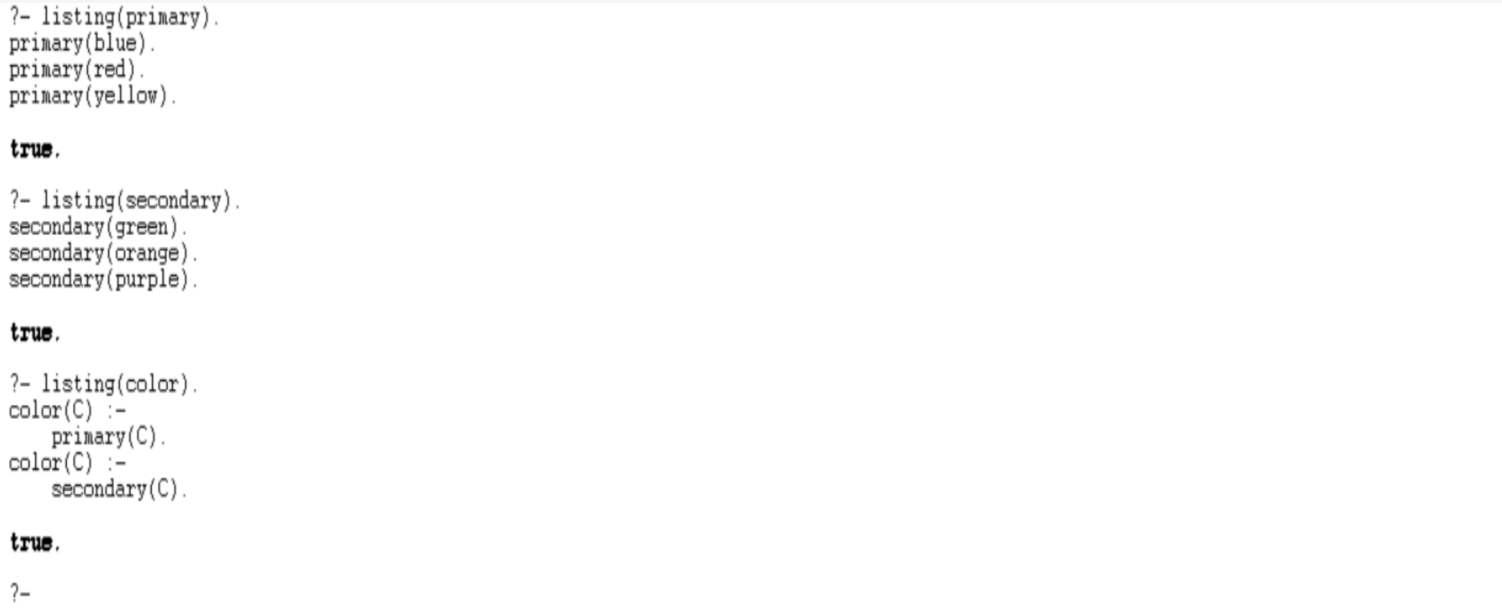


```
% -----------------------------------------------------------------------
% File: colors.pro
% Line: Six color facts, structured into primaries and secondaries
% -----------------------------------------------------------------------
% primary(P) :: P is a primary color
primary(blue).
primary(red).
primary(yellow).
% -----------------------------------------------------------------------
% primary(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:



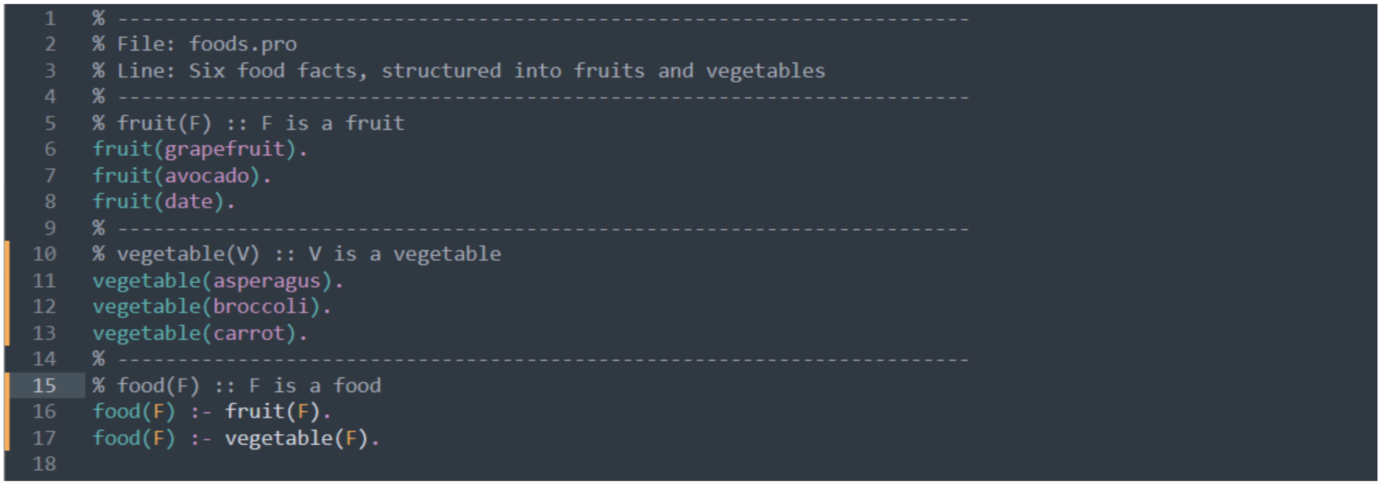
```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.0)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

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.

?-
```

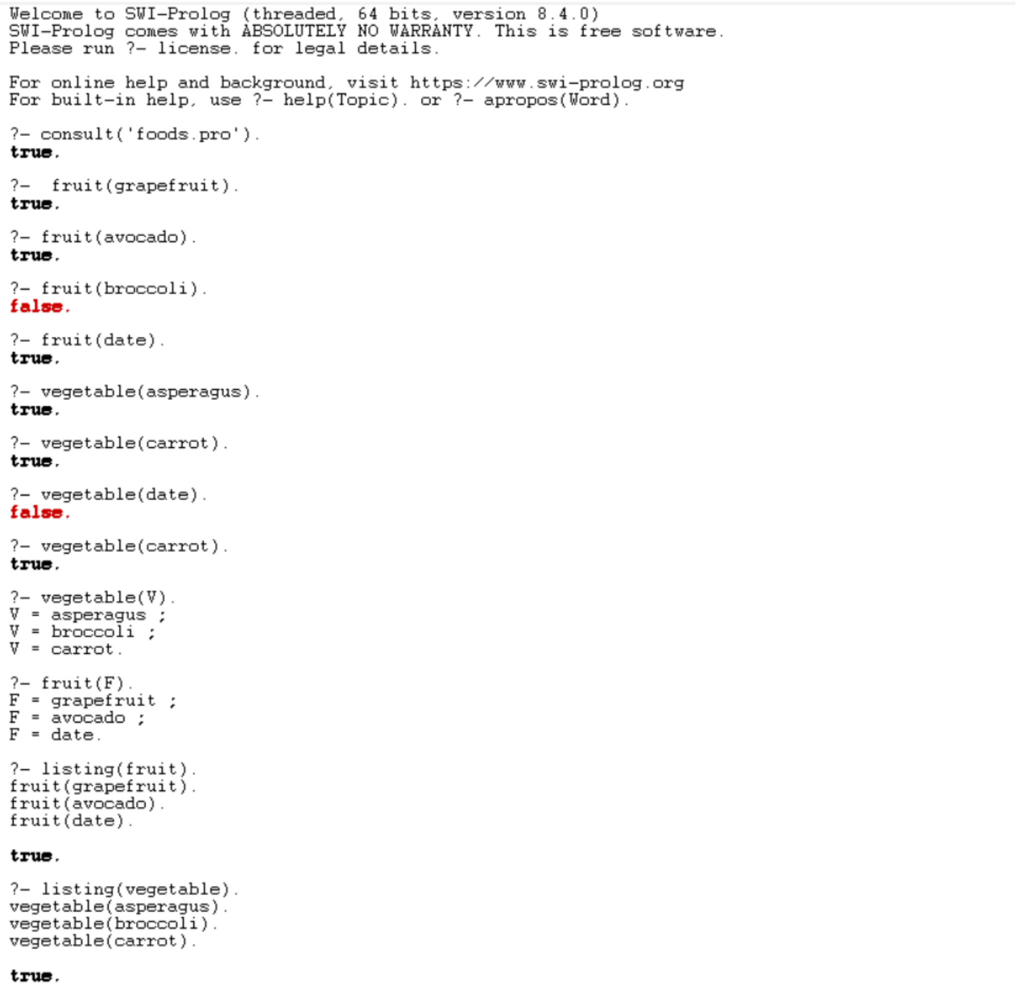
Task 2 - Food KB

Food KB Code:

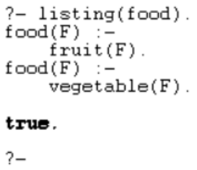


```
% ----------------------------------------------------------------------
% File: foods.pro
% Line: Six food facts, structured into fruits and vegetables
% ----------------------------------------------------------------------
% fruit(F) :: F is a fruit
fruit(grapefruit).
fruit(avocado).
fruit(date).
% ----------------------------------------------------------------------
% 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:



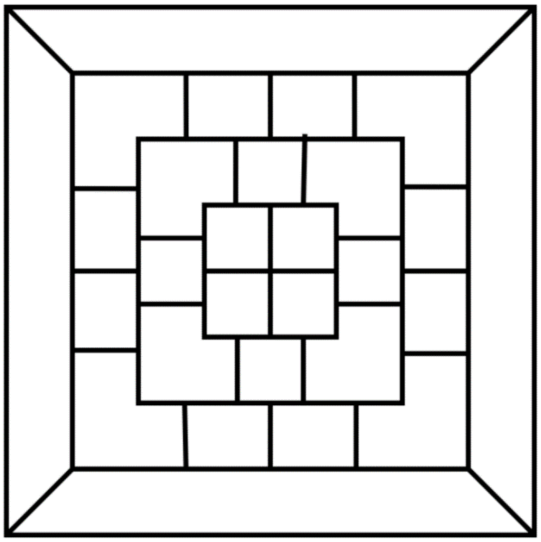
```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.0)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

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

?- consult('foods.pro').
true.

?-  fruit(grapefruit).
true.

?- fruit(avocado).
true.

?- fruit(broccoli).
false.

?- fruit(date).
true.

?- vegetable(asperagus).
true.

?- vegetable(carrot).
true.

?- vegetable(date).
false.

?- vegetable(carrot).
true.

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

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

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

true.

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

true.
```



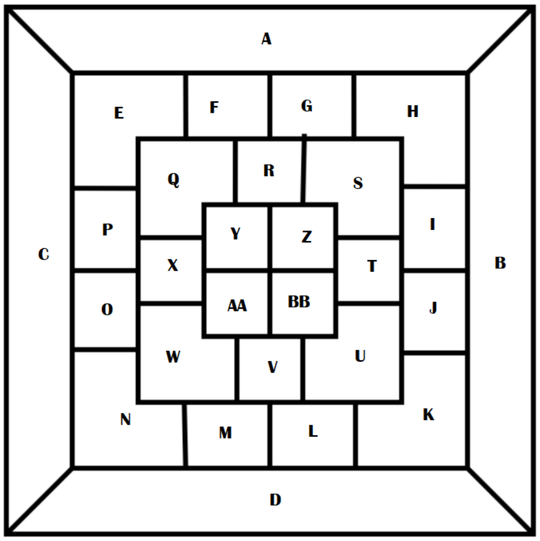
```
?- listing(food).
food(F) :-
    fruit(F).
food(F) :-
    vegetable(F).

true.

?-
```

Task 3 - Map Coloring

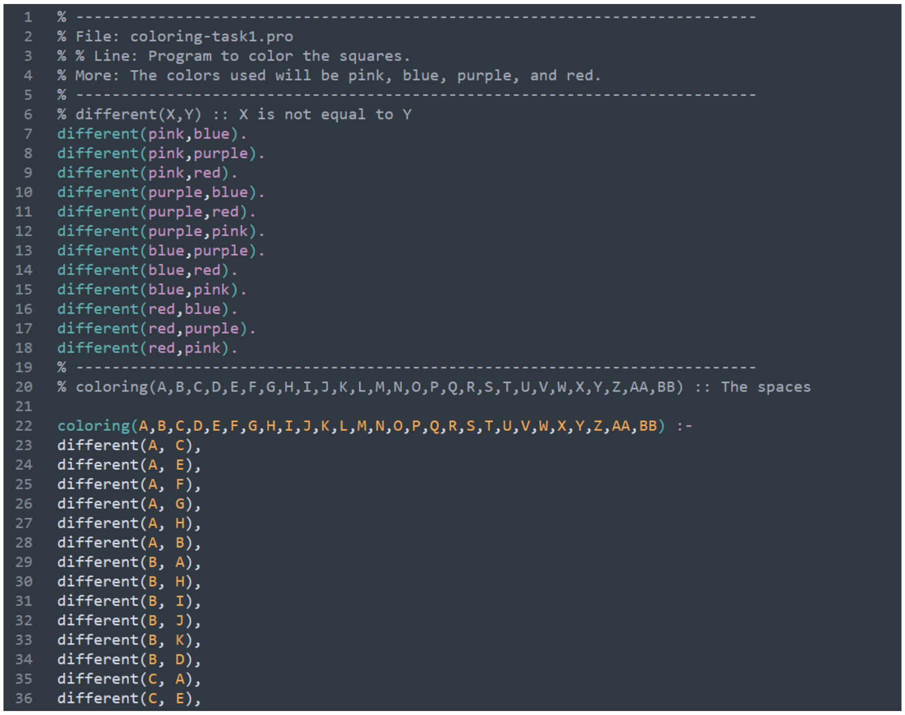
The Given Map:



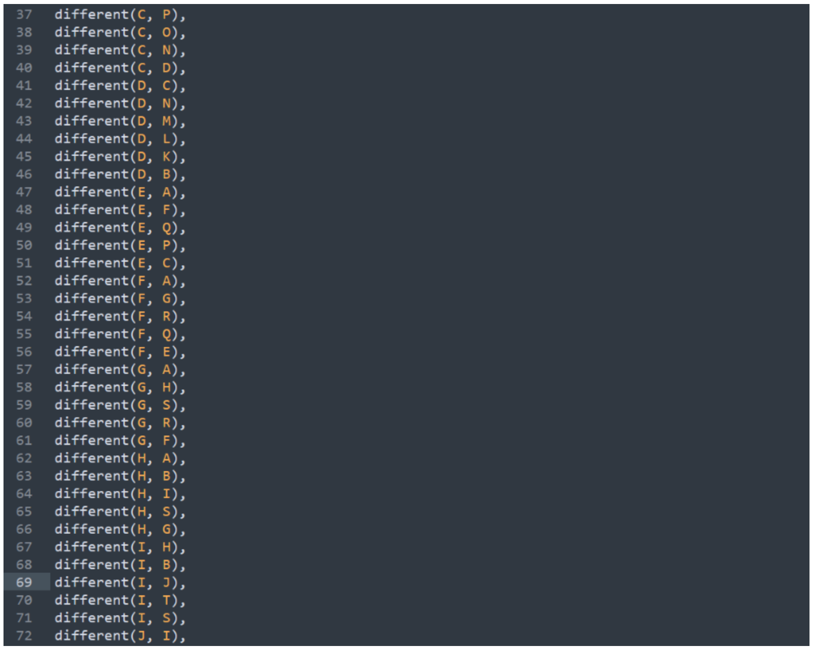
The Labeled Map:



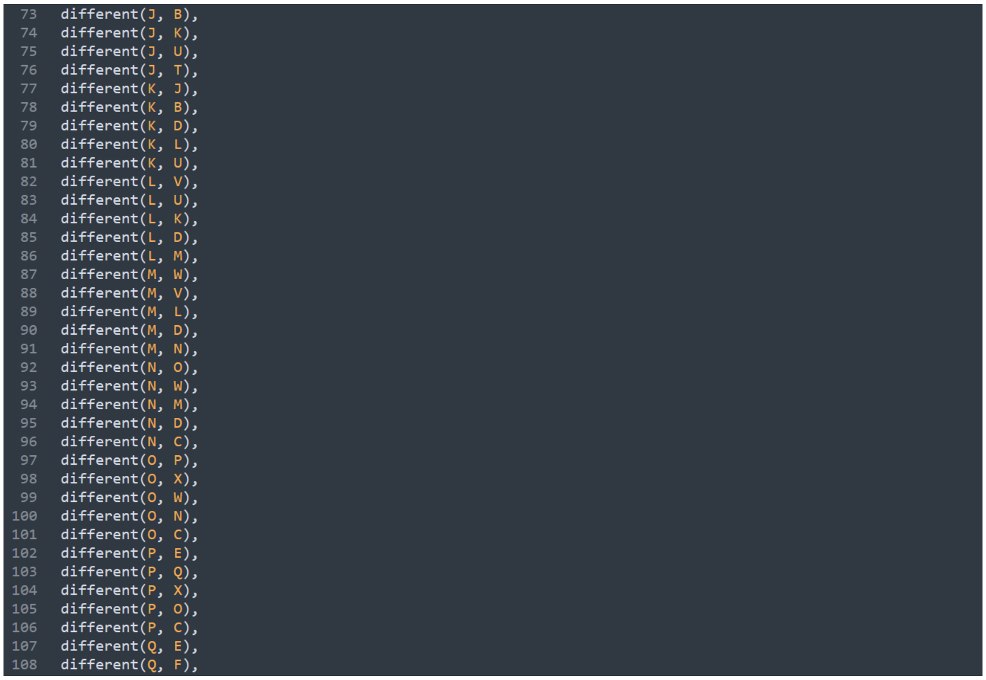
Code for Coloring the Map:



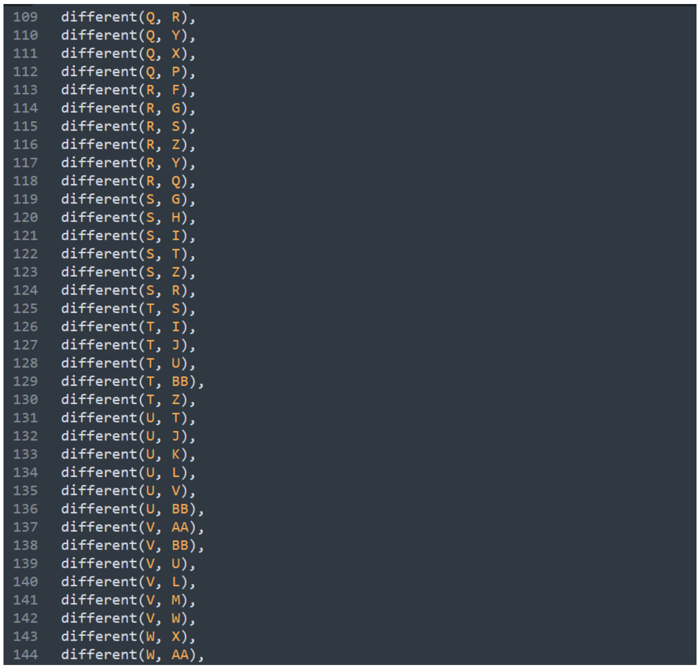
```prolog
% ---------------------------------------------------------------------------
% File: coloring-task1.pro
% % Line: Program to color the squares.
% More: The colors used will be pink, blue, purple, and red.
% ---------------------------------------------------------------------------
% different(X,Y) :: X is not equal to Y
different(pink,blue).
different(pink,purple).
different(pink,red).
different(purple,blue).
different(purple,red).
different(purple,pink).
different(blue,purple).
different(blue,red).
different(blue,pink).
different(red,blue).
different(red,purple).
different(red,pink).
% ---------------------------------------------------------------------------
% 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) :: The spaces

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, C),
different(A, E),
different(A, F),
different(A, G),
different(A, H),
different(A, B),
different(B, A),
different(B, H),
different(B, I),
different(B, J),
different(B, K),
different(B, D),
different(C, A),
different(C, E),
```



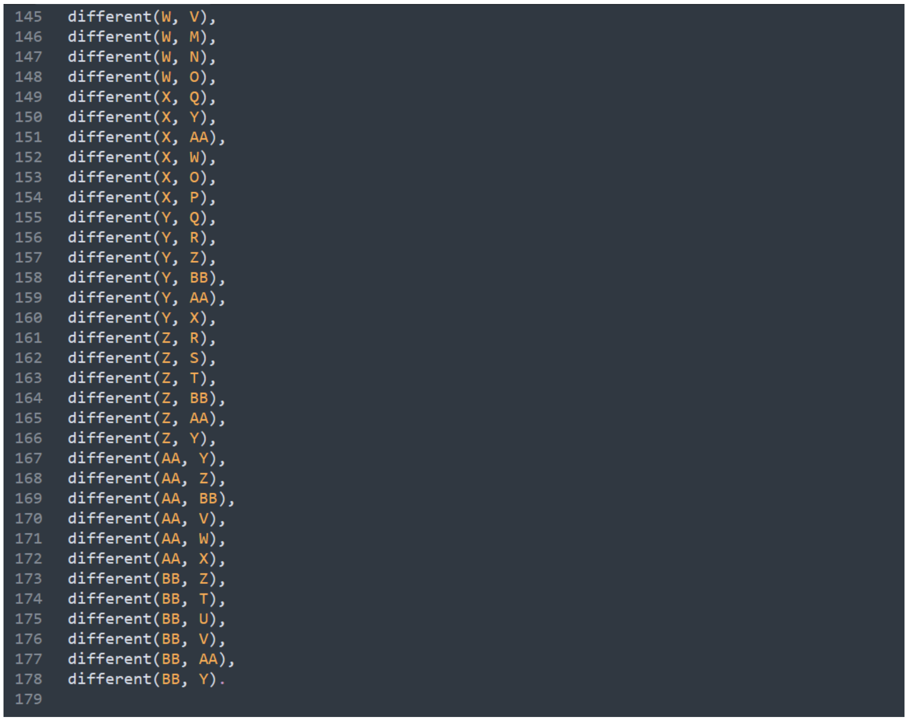
```prolog
different(C, P),
different(C, O),
different(C, N),
different(C, D),
different(D, C),
different(D, N),
different(D, M),
different(D, L),
different(D, K),
different(D, B),
different(E, A),
different(E, F),
different(E, Q),
different(E, P),
different(E, C),
different(F, A),
different(F, G),
different(F, R),
different(F, Q),
different(F, E),
different(G, A),
different(G, H),
different(G, S),
different(G, R),
different(G, F),
different(H, A),
different(H, B),
different(H, I),
different(H, S),
different(H, G),
different(I, H),
different(I, B),
different(I, J),
different(I, T),
different(I, S),
different(J, I),
```



```
different(J, B),
different(J, K),
different(J, U),
different(J, T),
different(K, J),
different(K, B),
different(K, D),
different(K, L),
different(K, U),
different(L, V),
different(L, U),
different(L, K),
different(L, D),
different(L, M),
different(M, W),
different(M, V),
different(M, L),
different(M, D),
different(M, N),
different(N, O),
different(N, W),
different(N, M),
different(N, D),
different(N, C),
different(O, P),
different(O, X),
different(O, W),
different(O, N),
different(O, C),
different(P, E),
different(P, Q),
different(P, X),
different(P, O),
different(P, C),
different(Q, E),
different(Q, F),
```

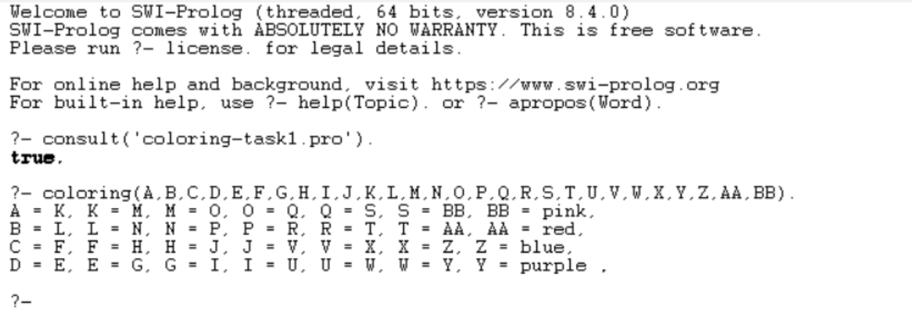


```
different(Q, R),
different(Q, Y),
different(Q, X),
different(Q, P),
different(R, F),
different(R, G),
different(R, S),
different(R, Z),
different(R, Y),
different(R, Q),
different(S, G),
different(S, H),
different(S, I),
different(S, T),
different(S, Z),
different(S, R),
different(T, S),
different(T, I),
different(T, J),
different(T, U),
different(T, BB),
different(T, Z),
different(U, T),
different(U, J),
different(U, K),
different(U, L),
different(U, V),
different(U, BB),
different(V, AA),
different(V, BB),
different(V, U),
different(V, L),
different(V, M),
different(V, W),
different(W, X),
different(W, AA),
```



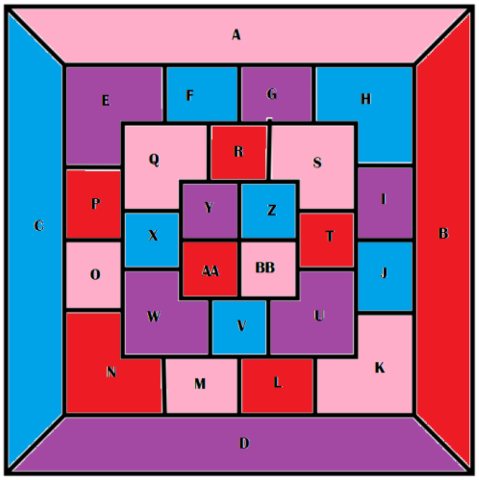
```
different(W, V),
different(W, M),
different(W, N),
different(W, O),
different(X, Q),
different(X, Y),
different(X, AA),
different(X, W),
different(X, O),
different(X, P),
different(Y, Q),
different(Y, R),
different(Y, Z),
different(Y, BB),
different(Y, AA),
different(Y, X),
different(Z, R),
different(Z,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

Map Coloring Interaction:



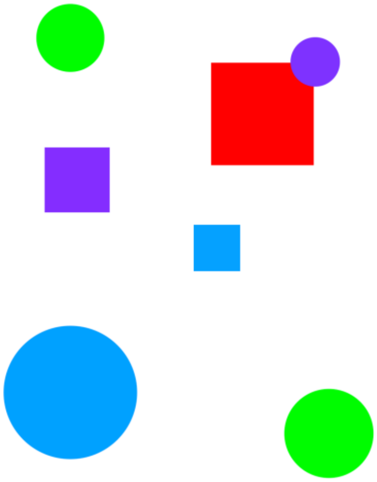
```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.0)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

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

?- consult('coloring-task1.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,BB).
A = K, K = M, M = O, O = Q, Q = S, S = BB, BB = pink,
B = L, L = N, N = P, P = R, R = T, T = AA, AA = red,
C = F, F = H, H = J, J = V, V = X, X = Z, Z = blue,
D = E, E = G, G = I, I = U, U = W, W = Y, Y = purple .

?-
```

The Colored Map:

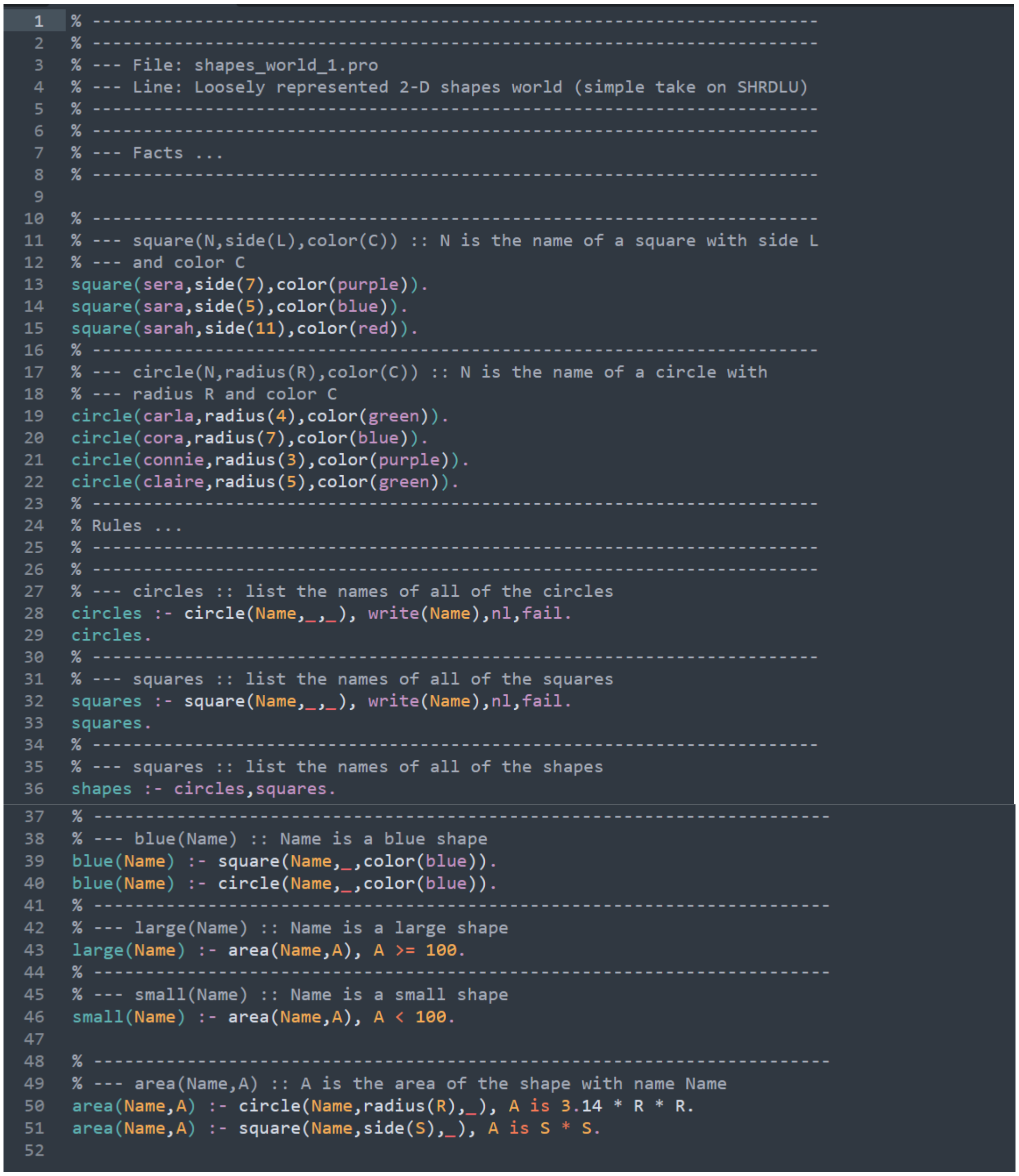


Task 4 - Floating Shapes World KB

Floating Shapes World Image:

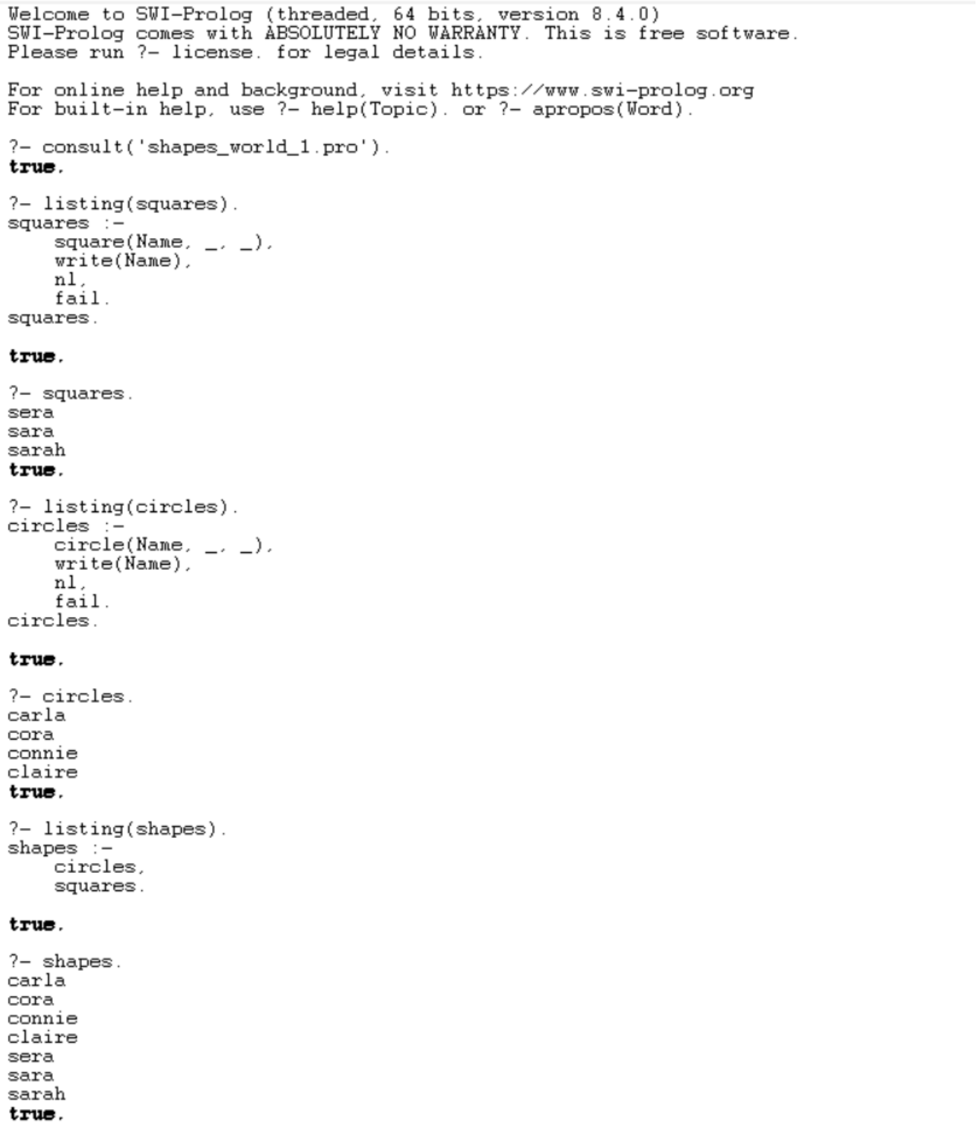


Floating Shapes World KB Code:

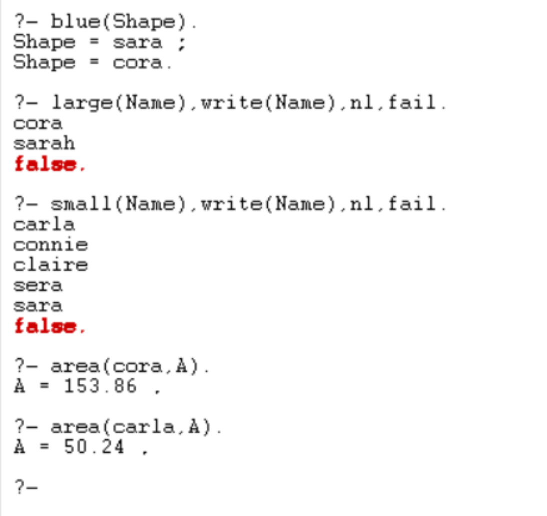


```
% -----------------------------------------------------------------------
% -----------------------------------------------------------------------
% --- File: shapes_world_1.pro
% --- Line: Loosely represented 2-D shapes world (simple take on SHRDLU)
% -----------------------------------------------------------------------
% -----------------------------------------------------------------------
% --- 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:



```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.0)
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).

?- 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 .

?-
```