Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

Learning Abstract

This assignment is intended to be an introduction to the logic-based programming language Prolog (short for programmable logic). Prolog consists of establishing a knowledge based with facts and rule in which the program is then loaded into a terminal in which queries are performed to determine the outcome of said queries which may be lists, true/false statements or even arithmetic evaluation. Prolog has been know to be used in the realm of artificial development.

Task 1: Map Coloring

Initial Image with Labeled Sections

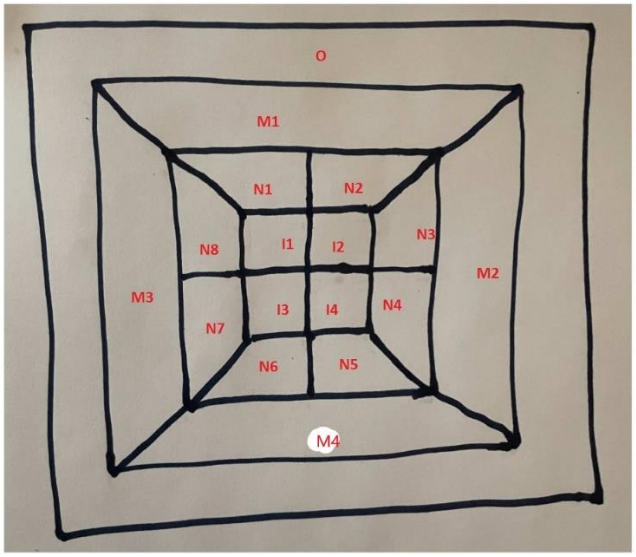
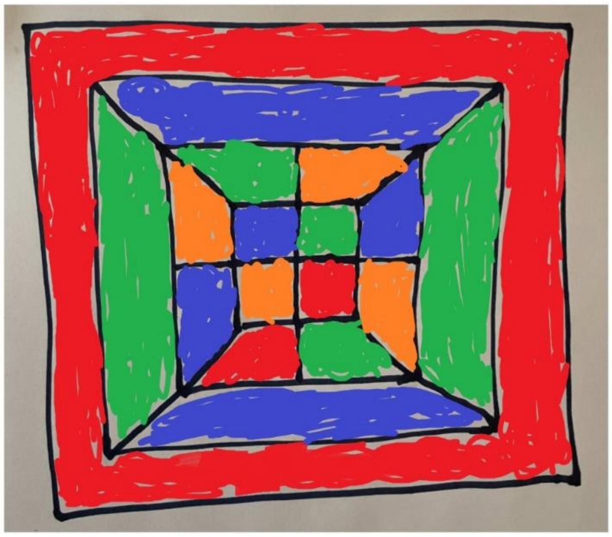
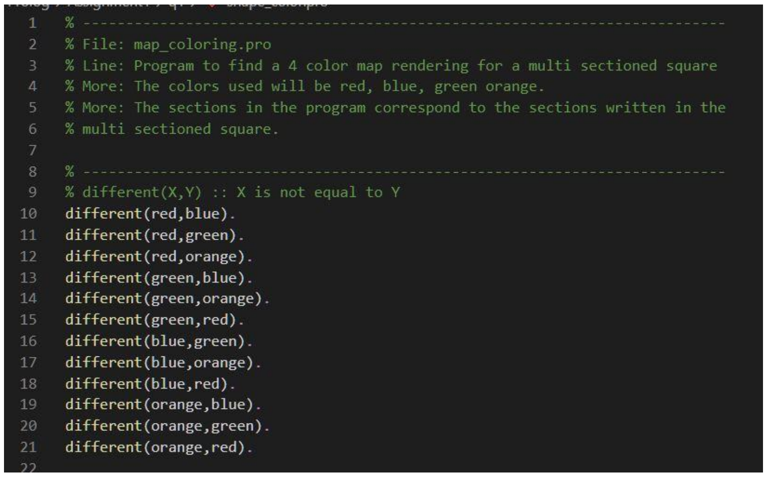


Image Colored as Patterned by Program

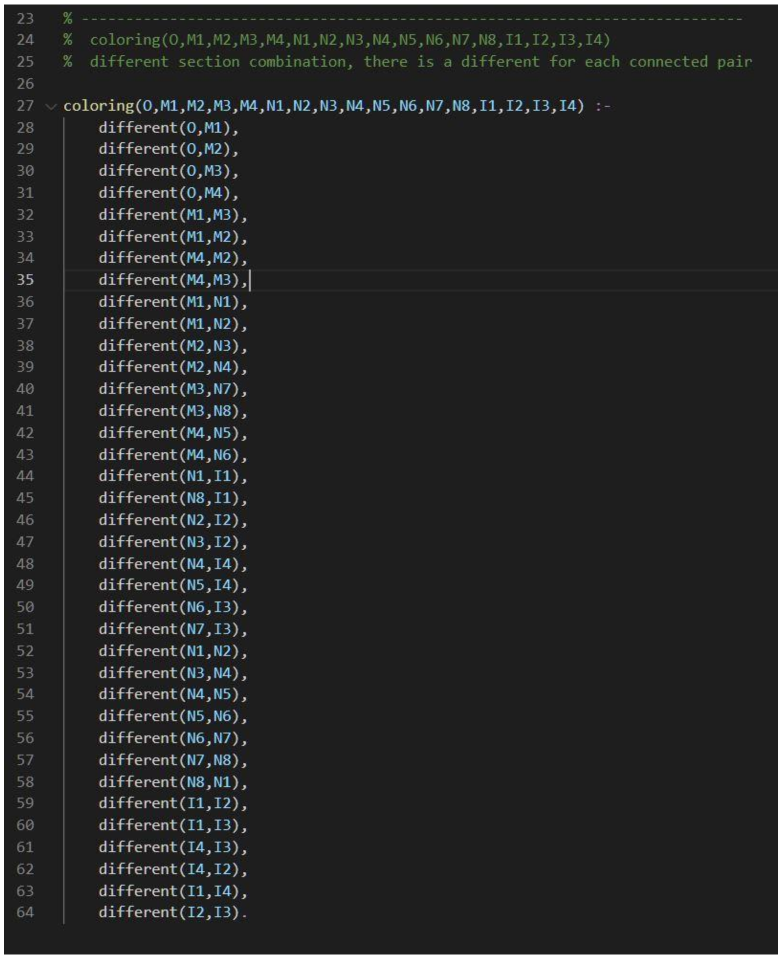


Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

Prolog Knowledge Base

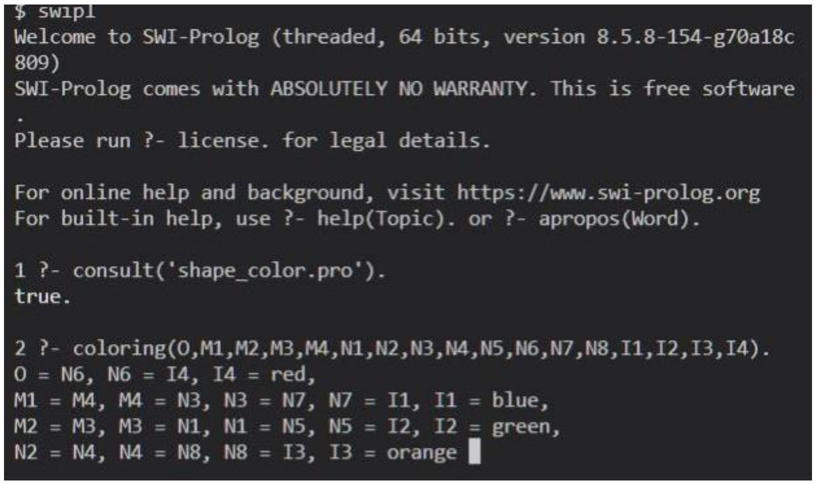


```
% -------------------------------------------------------------------------
% File: map_coloring.pro
% Line: Program to find a 4 color map rendering for a multi sectioned square
% More: The colors used will be red, blue, green orange.
% More: The sections in the program correspond to the sections written in the
% multi sectioned square.

% -------------------------------------------------------------------------
% 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(O,M1,M2,M3,M4,N1,N2,N3,N4,N5,N6,N7,N8,I1,I2,I3,I4)
%  different section combination, there is a different for each connected pair

coloring(O,M1,M2,M3,M4,N1,N2,N3,N4,N5,N6,N7,N8,I1,I2,I3,I4) :-
    different(O,M1),
    different(O,M2),
    different(O,M3),
    different(O,M4),
    different(M1,M3),
    different(M1,M2),
    different(M4,M2),
    different(M4,M3),
    different(M1,N1),
    different(M1,N2),
    different(M2,N3),
    different(M2,N4),
    different(M3,N7),
    different(M3,N8),
    different(M4,N5),
    different(M4,N6),
    different(N1,I1),
    different(N8,I1),
    different(N2,I2),
    different(N3,I2),
    different(N4,I4),
    different(N5,I4),
    different(N6,I3),
    different(N7,I3),
    different(N1,N2),
    different(N3,N4),
    different(N4,N5),
    different(N5,N6),
    different(N6,N7),
    different(N7,N8),
    different(N8,N1),
    different(I1,I2),
    different(I1,I3),
    different(I4,I3),
    different(I4,I2),
    different(I1,I4),
    different(I2,I3).
```

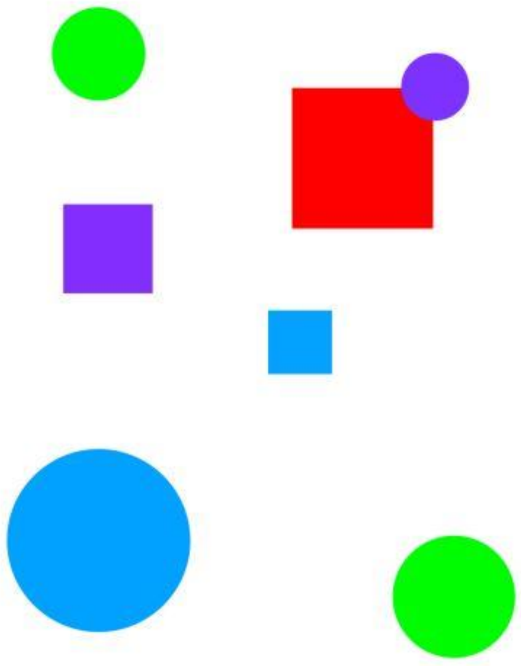
Demo Interaction Session



```
$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.5.8-154-g70a18c
809)
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).

1 ?- consult('shape_color.pro').
true.

2 ?- coloring(O,M1,M2,M3,M4,N1,N2,N3,N4,N5,N6,N7,N8,I1,I2,I3,I4).
O = N6, N6 = I4, I4 = red,
M1 = M4, M4 = N3, N3 = N7, N7 = I1, I1 = blue,
M2 = M3, M3 = N1, N1 = N5, N5 = I2, I2 = green,
N2 = N4, N4 = N8, N8 = I3, I3 = orange
```

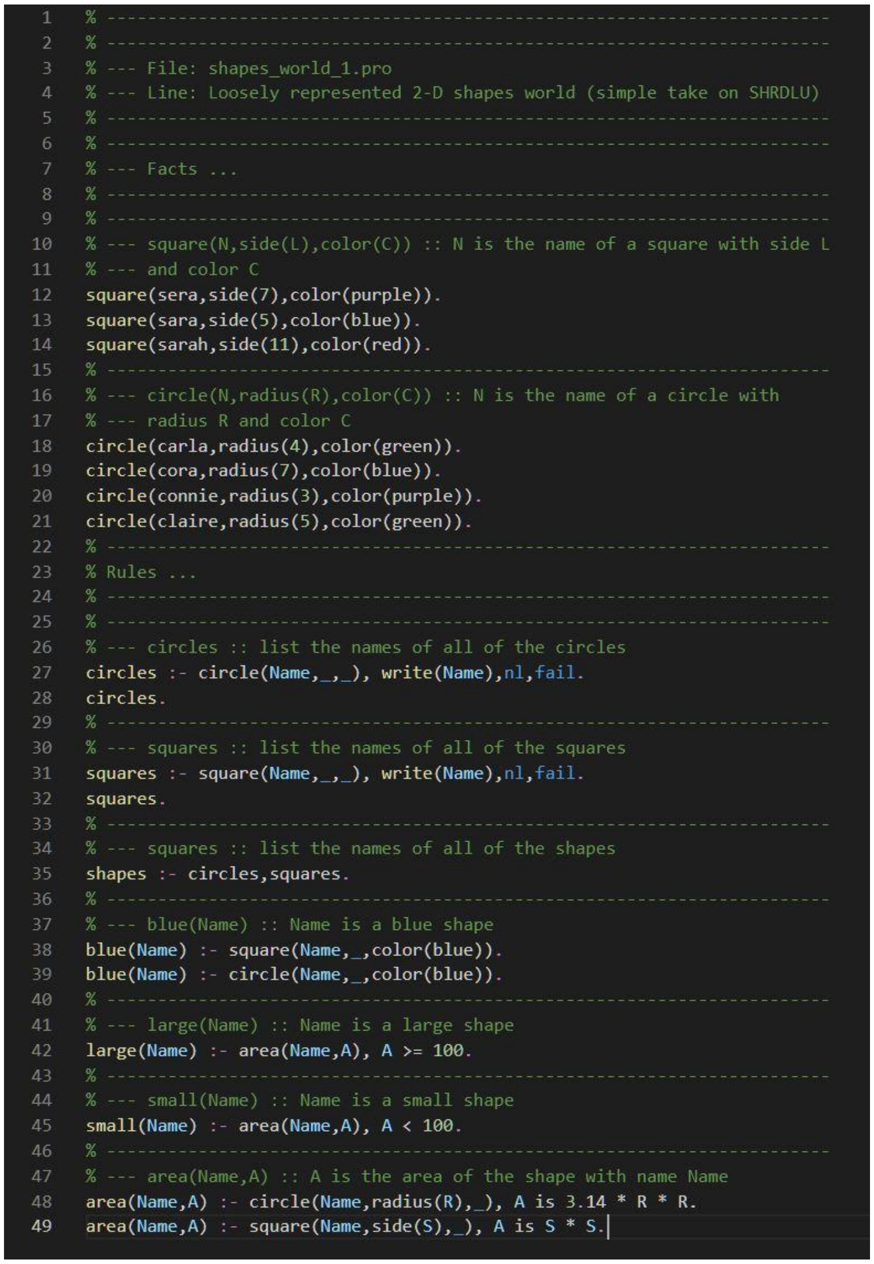
Task 2: The Floating Shapes World

Image Representation



Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

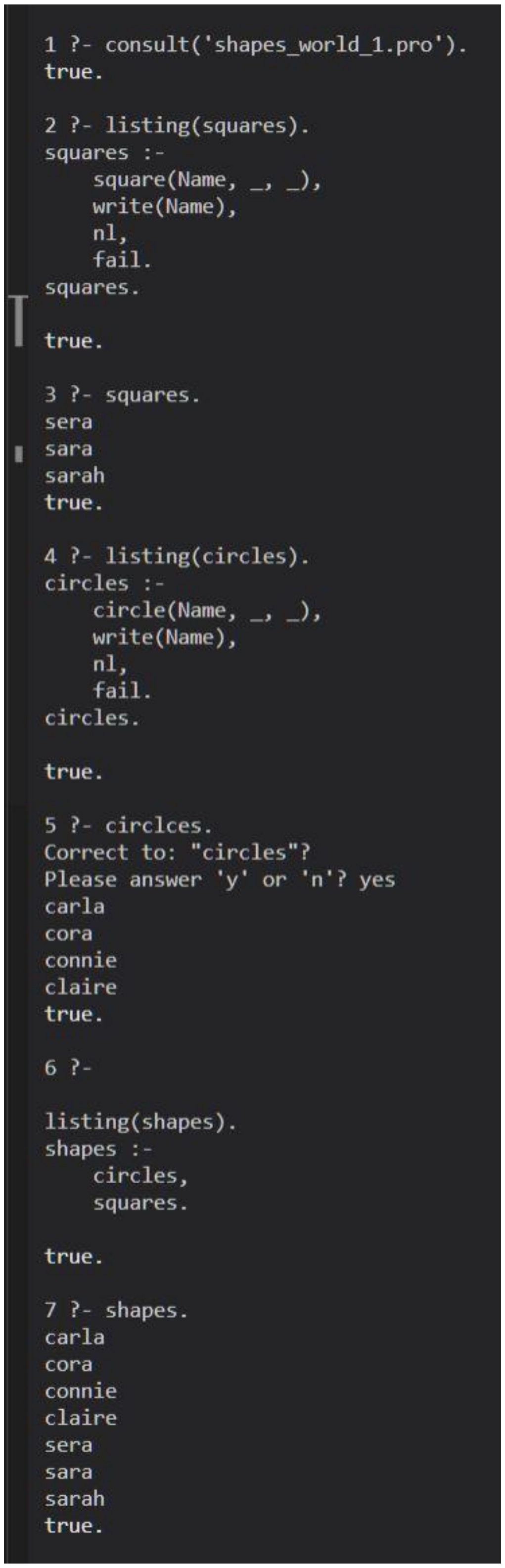
Prolog Knowledge Base



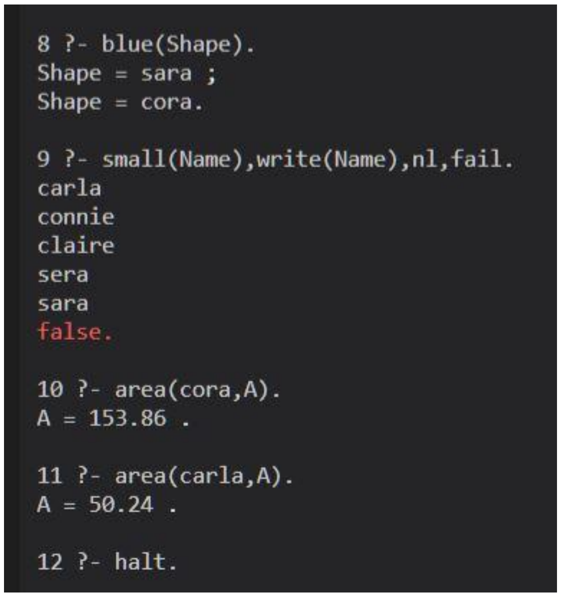
```
% ---------------------------------------------------------------------
% ---------------------------------------------------------------------
% --- File: shapes_world_1.pro
% --- Line: Loosely represented 2-D shapes world (simple take on SHRDLU)
% ---------------------------------------------------------------------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

Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

Demo Interaction Session



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

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

true.

3 ?- squares.
sera
sara
sarah
true.

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

true.

5 ?- circlces.
Correct to: "circles"?
Please answer 'y' or 'n'? yes
carla
cora
connie
claire
true.

6 ?-

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

true.

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



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

9 ?- small(Name),write(Name),nl,fail.
carla
connie
claire
sera
sara
false.

10 ?- area(cora,A).
A = 153.86 .

11 ?- area(carla,A).
A = 50.24 .

12 ?- halt.
```

Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

Task 3:

Determine Query Solution and Demo

```
%% Query 1: Is picachu a “creatio ex nihilo” (created out of nothing) pokemon? %%
?- cen(pikachu).
true.

%% Query 2: Is raichu a “creatio ex nihilo” pokemon? %%
?- cen(riachu).
false.

%$ Query 3: By means of hand intervention, list all of the “creatio ex nihilo”
pokemon.  %%
?- cen(Name).
Name = pikachu ;
Name = bulbasaur ;
Name = caterpie ;
Name = charmander ;
Name = vulpix ;
Name = poliwag ;
Name = squirtle ;
Name = staryu.


%% By means of the standard idiom of repetition, list all of the “creatio ex
nihilo” pokemon. %%
?- cen(Name),write(Name),nl,fail.
pikachu
bulbasaur
caterpie
charmander
vulpix
poliwag
squirtle
staryu
false.

%%  Does squirtle evolve into wartortle?
?- evolves(squirtle,wartortle).
true.

%%  Does wartortle evolve into squirtle? %%
?- evolves(wartortle,squirtle).
false.
```

Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

```
%% Does squirtle evolve into blastoise? %%
?- evolves(squirtle,blastoise).
false.

%% : By means of hand intervention, list all triples of pokemon such %%
%% that the first evolves into the second %%
%% and the second evolves into the third. %%
?- evolves(P,Q),evolves(Q,R).
X = bulbasaur,
Y = ivysaur,
Z = venusaur ;
X = caterpie,
Y = metapod,
Z = butterfree ;
X = charmander,
Y = charmeleon,
Z = charizard ;
X = poliwag,
Y = poliwhirl,
Z = poliwrath ;
X = squirtle,
Y = wartortle,
Z = blastoise ;
false.

%% By means of the standard idiom of repetition, list all pairs of pokemon such
that the first evolves %%
%% through an intermediary to the second - placing an arrow between each pair.%%
?-evolves(P,Q),evolves(Q,R),write(P),write('-->'),write(R),
bulbasaur --> venusaur
caterpie --> butterfree
charmander --> charizard
poliwag --> poliwrath
squirtle --> blastoise
false.

%% : By means of hand intervention, list all triples of pokemon such that the
first evolves into the second
%% and the second evolves into the third. %%
?- pokemon(name(N),_,_,_),write(N),nl,fail.
pikachu
raichu
```

```
bulbasaur
ivysaur
venusaur
caterpie
metapod
butterfree
charmander
charmeleon
charizard
vulpix
ninetails
poliwag
poliwhirl
polywrath
squirtle
wartortle
blastoise
staryu
starmie
false.

%% By means of the standard idiom of repetition, list the names of all of the
fire pokemon
?- pokemon(name(N),fire,_,_),write(N),nl,fail.
charmander
charmeleon
charizard
vulpix
ninetails
false.

%% By means of the standard idiom of repetition, provide a summary of each
pokemon and its kind,
%% representing each pairing of name and kind in the manner suggested by the
redacted demo.
?-
pokemon(name(Name),T,_,_),write('nks(name('),write(Name),write('),kind('),write(T
),write('))'),nl,fail.
nks(name(pikachu),kind(electric))
nks(name(raichu),kind(electric))
nks(name(bulbasaur),kind(grass))
nks(name(ivysaur),kind(grass))
nks(name(venusaur),kind(grass))
nks(name(caterpie),kind(grass))
nks(name(metapod),kind(grass))
nks(name(butterfree),kind(grass))
nks(name(charmander),kind(fire))
nks(name(charmeleon),kind(fire))
nks(name(charizard),kind(fire))
nks(name(vulpix),kind(fire))
nks(name(ninetails),kind(fire))
nks(name(poliwag),kind(water))
nks(name(poliwhirl),kind(water))
nks(name(polywrath),kind(water))
nks(name(squirtle),kind(water))
nks(name(wartortle),kind(water))
nks(name(blastoise),kind(water))
nks(name(staryu),kind(water))
nks(name(starmie),kind(water))
false.

%% What is the name of the pokemon with the waterfall attack?
?- pokemon(name(Name),_,_,attack(waterfall,_)).
N = wartortle

%% What is the name of the pokemon with the poison-powder attack?
?- pokemon(name(Name),_,_,attack(poison-powder,_)).
N = venusaur

%% By means of the standard idiom of repetition, list the names of the attacks of
all of the water pokemon.
?- pokemon(_,water,_,attack(Attack,_)),write(Attack),nl,fail.
water-gun
amnesia
dashing-punch
bubble
waterfall
hydro-pump
slap
star-freeze
false.

%% How much damage (hp count) can poliwhirl absorb?
?- pokemon(name(poliwhirl),_,hp(HP),_).
HP = 80
```

Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

```
%% How much damage (hp count) can butterfree absorb?
?- pokemon(name(butterfree),_,hp(HP),_).
HP = 130

%%  By means of the standard idiom of repetition, list the names of all of the
pokemon that can absorb
%% more than 85 units of damage
?- pokemon(name(Name),_,hp(HP),_), HP > 85, write(Name),nl,fail.
raichu
venusaur
butterfree
charizard
ninetails
polywrath
blastoise
false.

%% By means of the standard idiom of repetition, list the names of all of the
pokemon that can dish
%% out more than 60 units of damage with one instance of their attack.
?- pokemon(_,_,_,attack(Name,DMG)), DMG>60,write(Name),nl,fail.
thunder-shock
poison-powder
whirlwind
royal-blaze
fire-blast
false.

%% By means of the standard idiom of repetition, list the names and the hit point
value for each of the
%% creation ex nihilo" pokemon, with the results formatted as the redacted demo
suggests.%%
?- cen(Name),pokemon(name(Name),_,hp(HP),_),write(Name),write(':
'),write(HP),nl,fail.
pikachu: 60
bulbasaur: 40
caterpie: 50
charmander: 50
vulpix: 60
poliwag: 60
squirtle: 40
staryu: 40
false.
```

Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

Knowledge Base + Expanded Programs

```
% ---------------------------------------------------------------------------
% ---------------------------------------------------------------------------
% --- File: pokemon.pro
% --- Line: Just a few facts about pokemon
% ---------------------------------------------------------------------------

% ---------------------------------------------------------------------------
% --- cen(P) :: Pokemon P was "creatio ex nihilo"

cen(pikachu).
cen(bulbasaur).
cen(caterpie).
cen(charmander).
cen(vulpix).
cen(poliwag).
cen(squirtle).
cen(staryu).

% ---------------------------------------------------------------------------
% --- evolves(P,Q) :: Pokemon P directly evolves to pokemon Q

evolves(pikachu,raichu).
evolves(bulbasaur,ivysaur).
evolves(ivysaur,venusaur).
evolves(caterpie,metapod).
evolves(metapod,butterfree).
evolves(charmander,charmeleon).
evolves(charmeleon,charizard).
evolves(vulpix,ninetails).
evolves(poliwag,poliwhirl).
evolves(poliwhirl,poliwrath).
evolves(squirtle,wartortle).
evolves(wartortle,blastoise).
evolves(staryu,starmie).

% ---------------------------------------------------------------------------
% --- pokemon(name(N),T,hp(H),attach(A,D)) :: There is a pokemon with
% --- name N, type T, hit point value H, and attach named A that does
% --- damage D.

pokemon(name(pikachu), electric, hp(60), attack(gnaw, 10)).
pokemon(name(raichu), electric, hp(90), attack(thunder-shock, 90)).
```

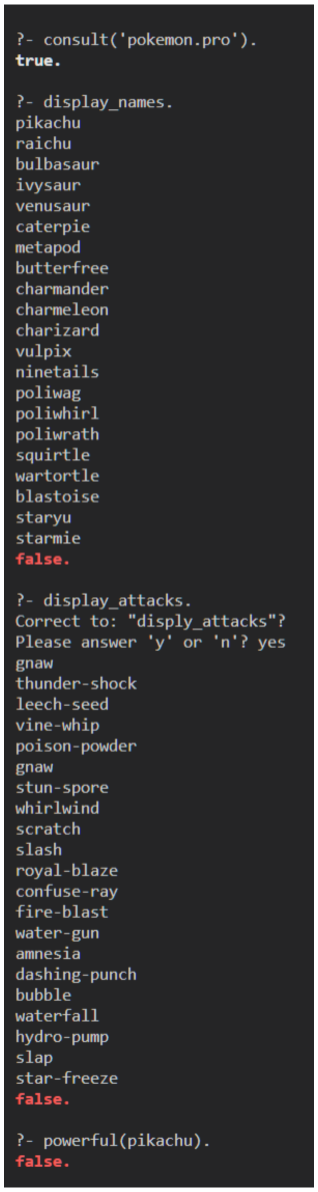
Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

```
pokemon(name(bulbasaur), grass, hp(40), attack(leech-seed, 20)).
pokemon(name(ivysaur), grass, hp(60), attack(vine-whip, 30)).
pokemon(name(venusaur), grass, hp(140), attack(poison-powder, 70)).

pokemon(name(caterpie), grass, hp(50), attack(gnaw, 20)).
pokemon(name(metapod), grass, hp(70), attack(stun-spore, 20)).
pokemon(name(butterfree), grass, hp(130), attack(whirlwind, 80)).

pokemon(name(charmander), fire, hp(50), attack(scratch, 10)).
pokemon(name(charmeleon), fire, hp(80), attack(slash, 50)).
pokemon(name(charizard), fire, hp(170), attack(royal-blaze, 100)).

pokemon(name(vulpix), fire, hp(60), attack(confuse-ray, 20)).
pokemon(name(ninetails), fire, hp(100), attack(fire-blast, 120)).

pokemon(name(poliwag), water, hp(60), attack(water-gun, 30)).
pokemon(name(poliwhirl), water, hp(80), attack(amnesia, 30)).
pokemon(name(poliwrath), water, hp(140), attack(dashing-punch, 50)).

pokemon(name(squirtle), water, hp(40), attack(bubble, 10)).
pokemon(name(wartortle), water, hp(80), attack(waterfall, 60)).
pokemon(name(blastoise), water, hp(140), attack(hydro-pump, 60)).

pokemon(name(staryu), water, hp(40), attack(slap, 20)).
pokemon(name(starmie), water, hp(60), attack(star-freeze, 20)).

display_names :- pokemon(name(Name),_,_,_),write(Name),nl,fail.

disply_attacks :- pokemon(_,_,_,attack(Attack,_)),write(Attack),nl,fail.

powerful(Name) :- pokemon(name(Name),_,_,attack(_,DMG)), DMG > 55.

tough(Name) :- pokemon(name(Name),_,hp(HP),_), HP > 100.

type(Name,T) :- pokemon(name(Name),T,_,_).

dump_kind(T) :- pokemon(name(Name), T, hp(HP), attack(Attack,DMG)),
    write('pokemon(name('), write(Name), write('), '), write(T),
    write(', hp('),write(HP),write('), attack('), write(Attack),
    write(', '), write(DMG), write(')).'),nl,fail.

display_cen :- cen(Pokemon),write(Pokemon),nl,fail.
```

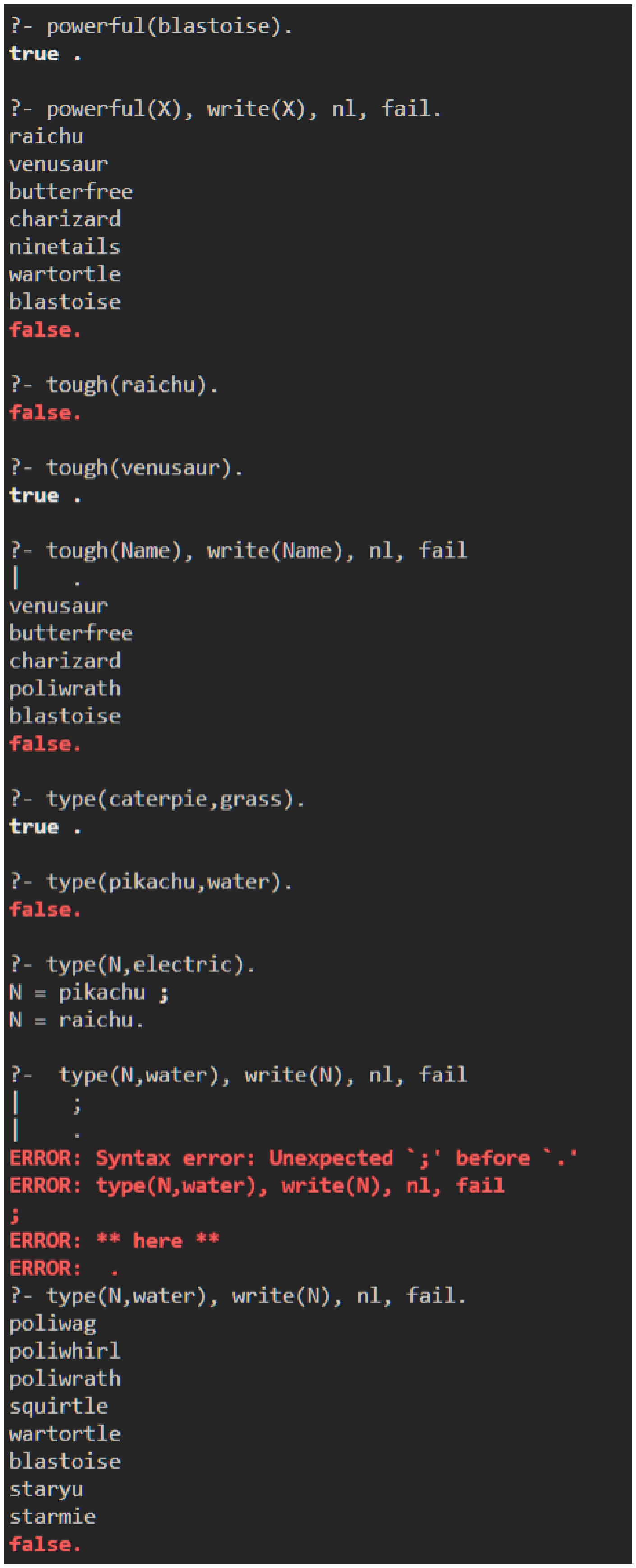
Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

```prolog
family(P) :- evolves(P,Q), evolve(Q,R), write(P), write(' '), write(Q),
    write(' '), write(R).
family(P) :- evolves(P,Q), \+ evolves(Q,R), write(P), write(' '), write(Q).

families :- cen(P), family(P), nl, fail.

lineage(P) :- evolves(P,Q), evolves(Q,R), display_info(P), nl,
    display_info(Q), nl, display_info(R).
lineage(P) :- evolves(P,Q), display_info(P), nl, display_info(Q).
lineage(P) :- display_info(P).

display_info(R) :- pokemon(name(Pokemon), T, hp(HP), attack(A, DMG)),
    write('pokemon(name('), write(Pokemon), write('), '),
    write(T), write(', hp('), write(HP), write('), attack('), write(A),
   write(', '), write(DMG), write(')), ').
```

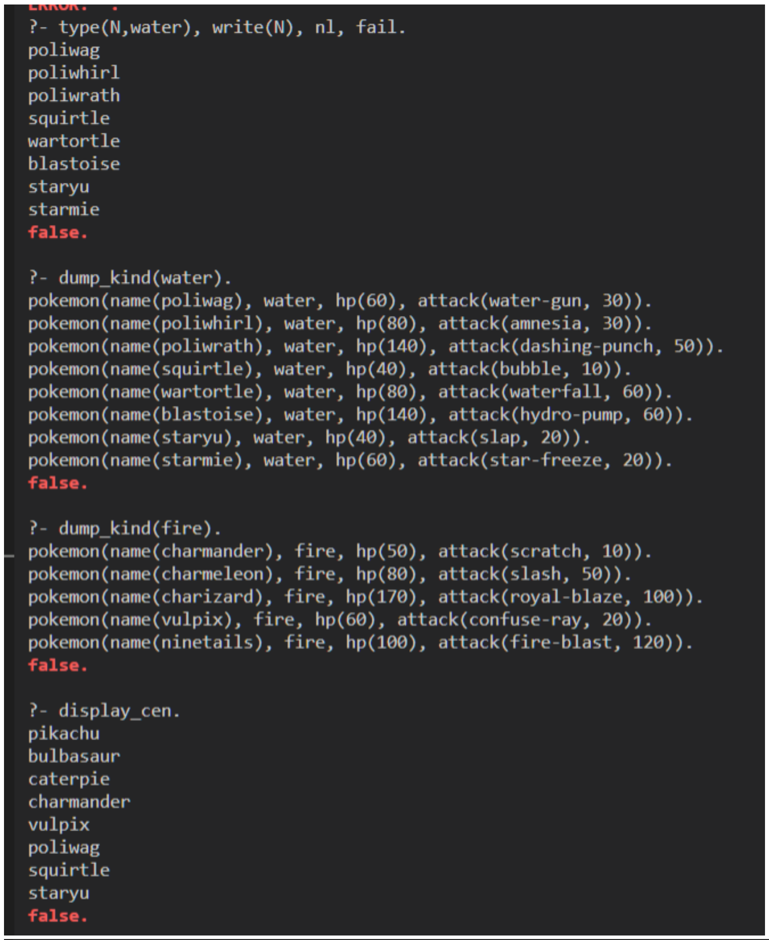
Expanded Program Demo



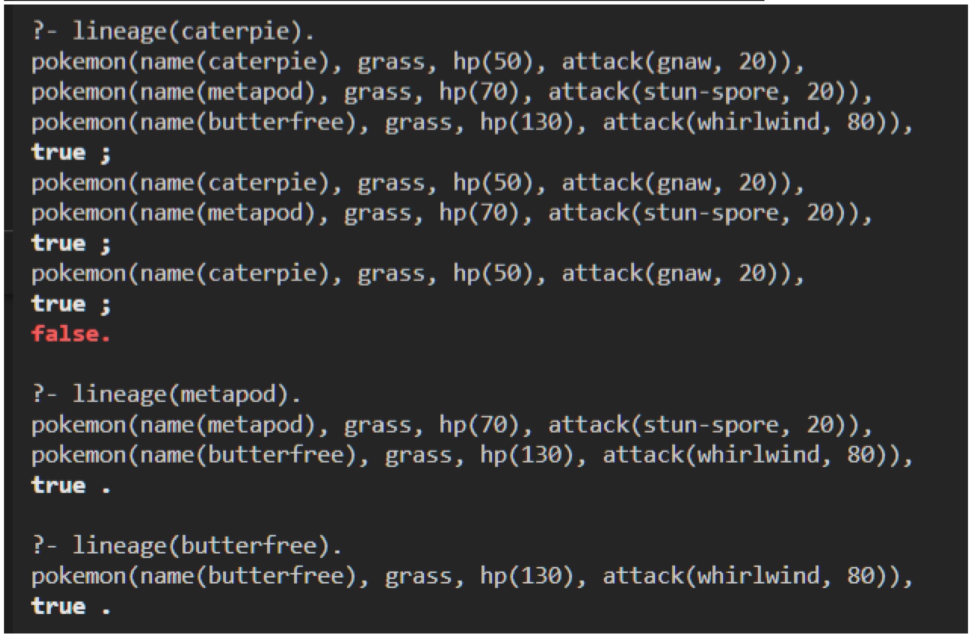
```
?- consult('pokemon.pro').
true.

?- display_names.
pikachu
raichu
bulbasaur
ivysaur
venusaur
caterpie
metapod
butterfree
charmander
charmeleon
charizard
vulpix
ninetails
poliwag
poliwhirl
poliwrath
squirtle
wartortle
blastoise
staryu
starmie
false.

?- display_attacks.
Correct to: "disply_attacks"?
Please answer 'y' or 'n'? yes
gnaw
thunder-shock
leech-seed
vine-whip
poison-powder
gnaw
stun-spore
whirlwind
scratch
slash
royal-blaze
confuse-ray
fire-blast
water-gun
amnesia
dashing-punch
bubble
waterfall
hydro-pump
slap
star-freeze
false.

?- powerful(pikachu).
false.
```



```
?- powerful(blastoise).
true .

?- powerful(X), write(X), nl, fail.
raichu
venusaur
butterfree
charizard
ninetails
wartortle
blastoise
false.

?- tough(raichu).
false.

?- tough(venusaur).
true .

?- tough(Name), write(Name), nl, fail
|    .
venusaur
butterfree
charizard
poliwrath
blastoise
false.

?- type(caterpie,grass).
true .

?- type(pikachu,water).
false.

?- type(N,electric).
N = pikachu ;
N = raichu.

?-  type(N,water), write(N), nl, fail
|    ;
|    .
ERROR: Syntax error: Unexpected `;' before `.'
ERROR: type(N,water), write(N), nl, fail
;
ERROR: ** here **
ERROR:  .
?- type(N,water), write(N), nl, fail.
poliwag
poliwhirl
poliwrath
squirtle
wartortle
blastoise
staryu
starmie
false.
```



```
?- type(N,water), write(N), nl, fail.
poliwag
poliwhirl
poliwrath
squirtle
wartortle
blastoise
staryu
starmie
false.

?- dump_kind(water).
pokemon(name(poliwag), water, hp(60), attack(water-gun, 30)).
pokemon(name(poliwhirl), water, hp(80), attack(amnesia, 30)).
pokemon(name(poliwrath), water, hp(140), attack(dashing-punch, 50)).
pokemon(name(squirtle), water, hp(40), attack(bubble, 10)).
pokemon(name(wartortle), water, hp(80), attack(waterfall, 60)).
pokemon(name(blastoise), water, hp(140), attack(hydro-pump, 60)).
pokemon(name(staryu), water, hp(40), attack(slap, 20)).
pokemon(name(starmie), water, hp(60), attack(star-freeze, 20)).
false.

?- dump_kind(fire).
pokemon(name(charmander), fire, hp(50), attack(scratch, 10)).
pokemon(name(charmeleon), fire, hp(80), attack(slash, 50)).
pokemon(name(charizard), fire, hp(170), attack(royal-blaze, 100)).
pokemon(name(vulpix), fire, hp(60), attack(confuse-ray, 20)).
pokemon(name(ninetails), fire, hp(100), attack(fire-blast, 120)).
false.

?- display_cen.
pikachu
bulbasaur
caterpie
charmander
vulpix
poliwag
squirtle
staryu
false.
```

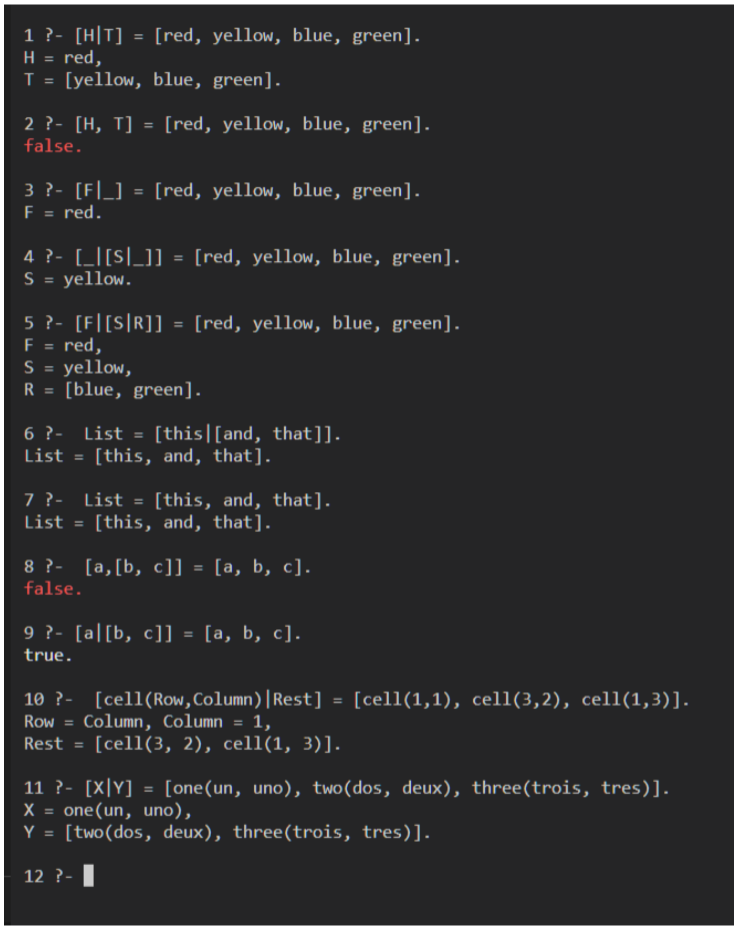


```
?- lineage(caterpie).
pokemon(name(caterpie), grass, hp(50), attack(gnaw, 20)),
pokemon(name(metapod), grass, hp(70), attack(stun-spore, 20)),
pokemon(name(butterfree), grass, hp(130), attack(whirlwind, 80)),
true ;
pokemon(name(caterpie), grass, hp(50), attack(gnaw, 20)),
pokemon(name(metapod), grass, hp(70), attack(stun-spore, 20)),
true ;
pokemon(name(caterpie), grass, hp(50), attack(gnaw, 20)),
true ;
false.

?- lineage(metapod).
pokemon(name(metapod), grass, hp(70), attack(stun-spore, 20)),
pokemon(name(butterfree), grass, hp(130), attack(whirlwind, 80)),
true .

?- lineage(butterfree).
pokemon(name(butterfree), grass, hp(130), attack(whirlwind, 80)),
true .
```

Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

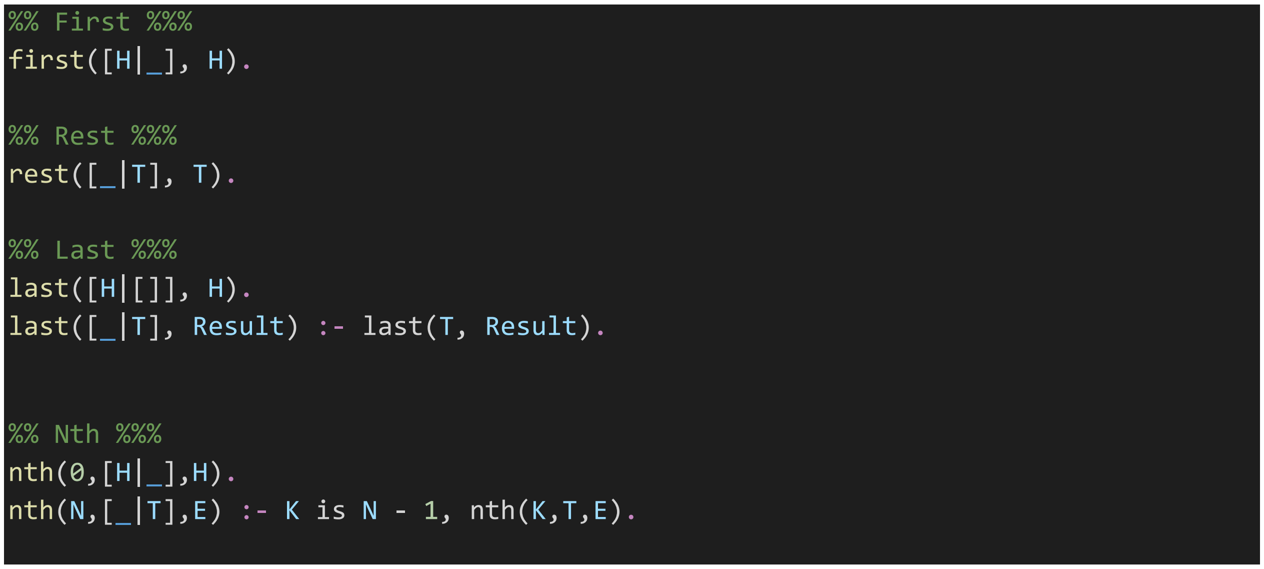
Task 4:

Part 1 - Demo – "Head/Tail Referencing Exercises"



```
1 ?- [H|T] = [red, yellow, blue, green].
H = red,
T = [yellow, blue, green].

2 ?- [H, T] = [red, yellow, blue, green].
false.

3 ?- [F|_] = [red, yellow, blue, green].
F = red.

4 ?- [_|[S|_]] = [red, yellow, blue, green].
S = yellow.

5 ?- [F|[S|R]] = [red, yellow, blue, green].
F = red,
S = yellow,
R = [blue, green].

6 ?-  List = [this|[and, that]].
List = [this, and, that].

7 ?-  List = [this, and, that].
List = [this, and, that].

8 ?-  [a,[b, c]] = [a, b, c].
false.

9 ?- [a|[b, c]] = [a, b, c].
true.

10 ?-  [cell(Row,Column)|Rest] = [cell(1,1), cell(3,2), cell(1,3)].
Row = Column, Column = 1,
Rest = [cell(3, 2), cell(1, 3)].

11 ?- [X|Y] = [one(un, uno), two(dos, deux), three(trois, tres)].
X = one(un, uno),
Y = [two(dos, deux), three(trois, tres)].

12 ?-
```

Part 2 - List_processors.pro



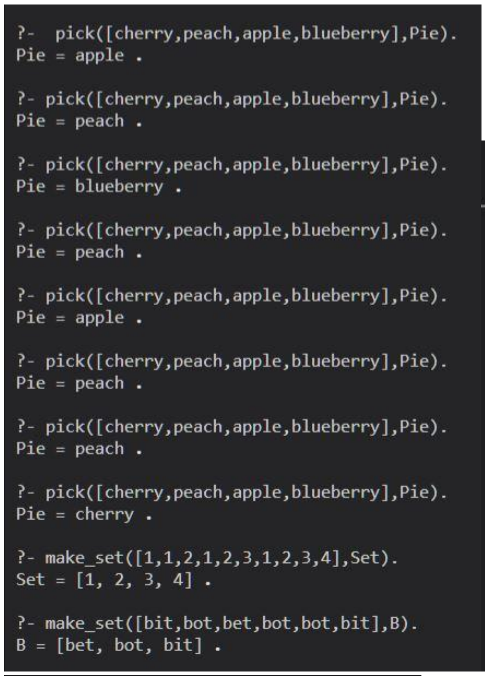
```
%% First %%%
first([H|_], H).

%% Rest %%%
rest([_|T], T).

%% Last %%%
last([H|[]], H).
last([_|T], Result) :- last(T, Result).


%% Nth %%%
nth(0,[H|_],H).
nth(N,[_|T],E) :- K is N - 1, nth(K,T,E).
```

Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

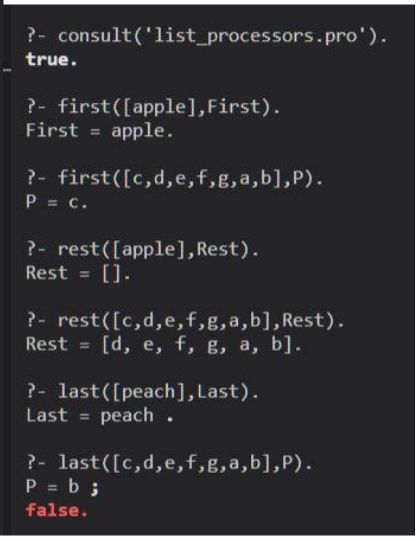
```
%% Writelist %%%
writelist([]).
writelist([H|T]) :-
    write(H), nl, writelist(T).

%% Sum %%%
sum([],0).
sum([Head|Tail],Sum) :-
    sum(Tail,SumOfTail),
    Sum is Head + SumOfTail.

%% Product %%%
product([],1).
product([Head|Tail],Product) :-
    product(Tail,ProductOfTail),
    Product is Head * ProductOfTail.

%% Addfirst %%%
add_first(X,L,[X|L]).

%% Add_last %%%
add_last(X,[],[X]).
add_last(X,[H|T],[H|TX]) :-
    add_last(X,T,TX).

%% Iota %%%
iota(0,[]).
iota(N,IotaN) :-
    K is N - 1,
    iota(K,IotaK),
    add_last(N,IotaK,IotaN).

%% Pick %%%
pick(L,Item) :-
    length(L,Length),
    random(0,Length,RN),
    nth(RN,L,Item).

%% Make_set %%%
make_set([],[]).
make_set([H|T],TS) :-
    member(H,T),
    make_set(T,TS).
make_set([H|T],[H|TS]) :-
    make_set(T,TS).

%% Make List %%
make_list(0,_,_).
make_list(N,E,L) :- K - 1,
    make_list(K,E,NL),
    add_last(E,NL,L).

%% But first %%
but_first(L,NL) :- rest(L,NL).

%% But Last %%
but_last([],[]).
but_last([_|[]],[]).
but_last([H|T],L) :-
    but_last(T,NL),
    add_first(H,NL,L).

%% Is palindrome %%
is_palindrome([]).
is_palindrome([_|[]],[]).
is_palindrome(L) :- first(L,First), last(L,Last),
    First = Last, but_first(L,NL),
    but_last(NL,NNL), is_palindrome(NNL).

%% noun_phrase %%
noun_phrase(NP) :-
    pick([yummy,funny,beautiful,talented,based,ridiculiouus],Adj),
    pick([player,ganster,gamer,writer,programmer,artist,dogwalker,babysitter],Nou
n),
    add_last(Adj, [the], L),
    add_last(Noun,L,NP).

%% sentence phrase %%
sentence(S) :-
    pick([fought,adminsatred,dressed,challenged,proctored,cut,ran],V),
    noun_phrase(NP),
    add_last(V,NP,NPV),
    noun_phrase(NP1),
    nth(0,NP1,E1),
    nth(1,NP1,E2),
    nth(2,NP1,E3),
    add_last(E1,NPV,NPV1),
    add_last(E2,NPV1,NPV2),
    add_last(E3,NPV2,S).
```

Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

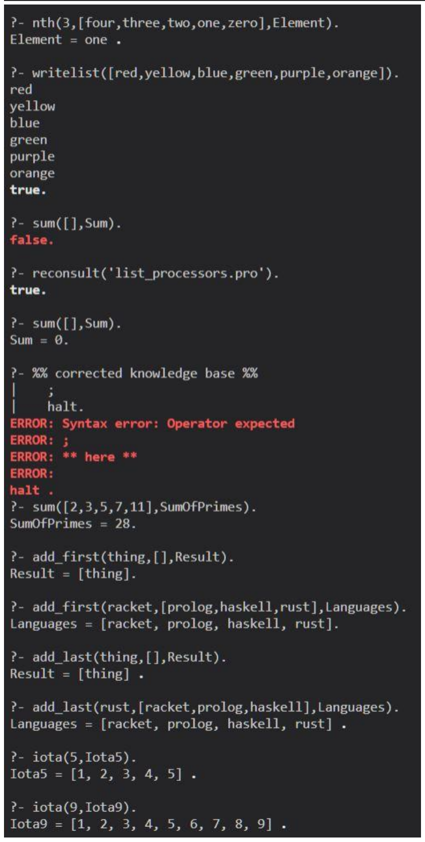
Part 3 – List processors demo interaction



```
?-  pick([cherry,peach,apple,blueberry],Pie).
Pie = apple .

?- pick([cherry,peach,apple,blueberry],Pie).
Pie = peach .

?- pick([cherry,peach,apple,blueberry],Pie).
Pie = blueberry .

?- pick([cherry,peach,apple,blueberry],Pie).
Pie = peach .

?- pick([cherry,peach,apple,blueberry],Pie).
Pie = apple .

?- pick([cherry,peach,apple,blueberry],Pie).
Pie = peach .

?- pick([cherry,peach,apple,blueberry],Pie).
Pie = peach .

?- pick([cherry,peach,apple,blueberry],Pie).
Pie = cherry .

?- make_set([1,1,2,1,2,3,1,2,3,4],Set).
Set = [1, 2, 3, 4] .

?- make_set([bit,bot,bet,bot,bot,bit],B).
B = [bet, bot, bit] .
```



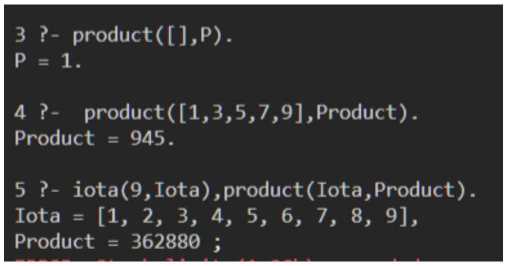
```
?- consult('list_processors.pro').
true.

?- first([apple],First).
First = apple.

?- first([c,d,e,f,g,a,b],P).
P = c.

?- rest([apple],Rest).
Rest = [].

?- rest([c,d,e,f,g,a,b],Rest).
Rest = [d, e, f, g, a, b].

?- last([peach],Last).
Last = peach .

?- last([c,d,e,f,g,a,b],P).
P = b ;
false.
```



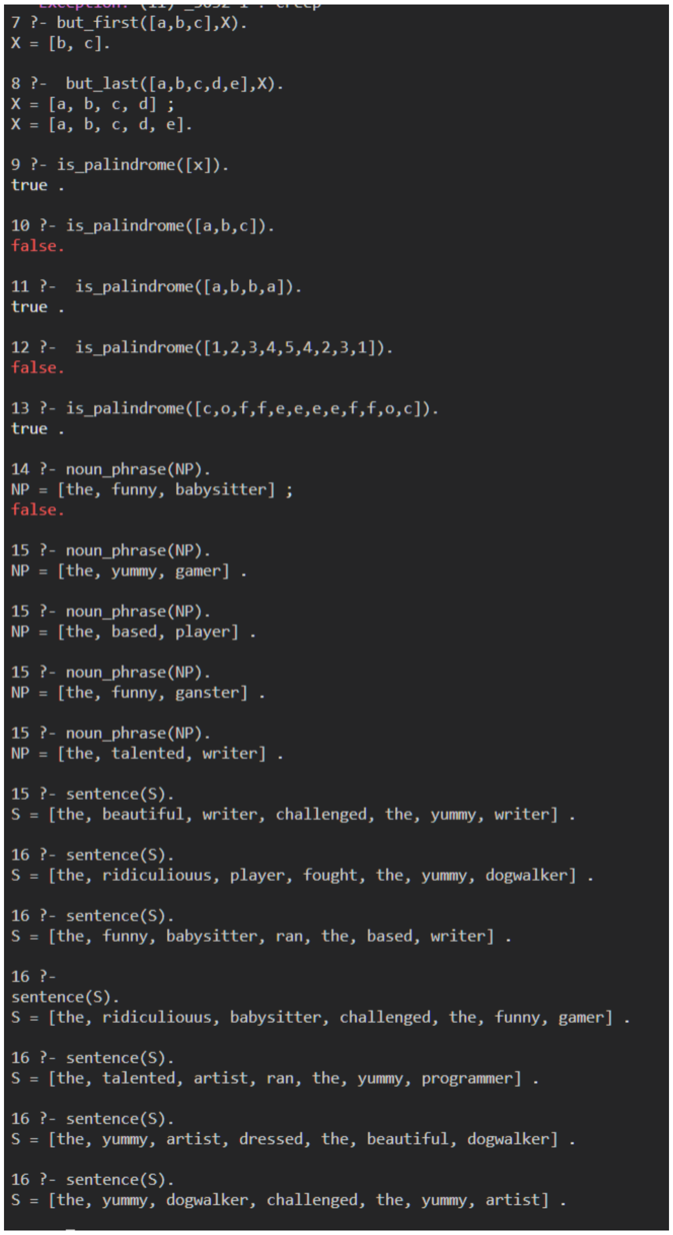
```
?- nth(3,[four,three,two,one,zero],Element).
Element = one .

?- writelist([red,yellow,blue,green,purple,orange]).
red
yellow
blue
green
purple
orange
true.

?- sum([],Sum).
false.

?- reconsult('list_processors.pro').
true.

?- sum([],Sum).
Sum = 0.

?- %% corrected knowledge base %%
|    ;
|    halt.
ERROR: Syntax error: Operator expected
ERROR: ;
ERROR: ** here **
ERROR:
halt .
?- sum([2,3,5,7,11],SumOfPrimes).
SumOfPrimes = 28.

?- add_first(thing,[],Result).
Result = [thing].

?- add_first(racket,[prolog,haskell,rust],Languages).
Languages = [racket, prolog, haskell, rust].

?- add_last(thing,[],Result).
Result = [thing] .

?- add_last(rust,[racket,prolog,haskell],Languages).
Languages = [racket, prolog, haskell, rust] .

?- iota(5,Iota5).
Iota5 = [1, 2, 3, 4, 5] .

?- iota(9,Iota9).
Iota9 = [1, 2, 3, 4, 5, 6, 7, 8, 9] .
```

Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344

Part 4 – List Processing Exercise

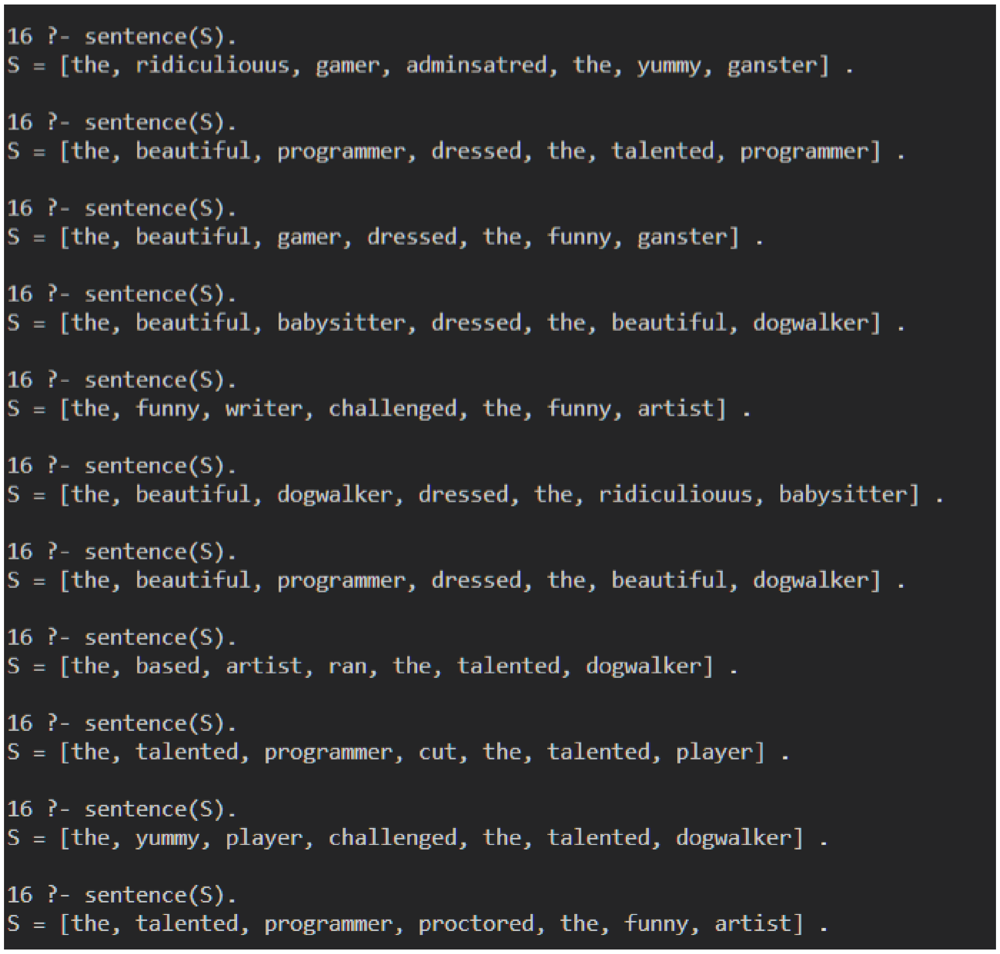


```
3 ?- product([],P).
P = 1.

4 ?-  product([1,3,5,7,9],Product).
Product = 945.

5 ?- iota(9,Iota),product(Iota,Product).
Iota = [1, 2, 3, 4, 5, 6, 7, 8, 9],
Product = 362880 ;
```



```
7 ?- but_first([a,b,c],X).
X = [b, c].

8 ?-  but_last([a,b,c,d,e],X).
X = [a, b, c, d] ;
X = [a, b, c, d, e].

9 ?- is_palindrome([x]).
true .

10 ?- is_palindrome([a,b,c]).
false.

11 ?-  is_palindrome([a,b,b,a]).
true .

12 ?-  is_palindrome([1,2,3,4,5,4,2,3,1]).
false.

13 ?- is_palindrome([c,o,f,f,e,e,e,e,f,f,o,c]).
true .

14 ?- noun_phrase(NP).
NP = [the, funny, babysitter] ;
false.

15 ?- noun_phrase(NP).
NP = [the, yummy, gamer] .

15 ?- noun_phrase(NP).
NP = [the, based, player] .

15 ?- noun_phrase(NP).
NP = [the, funny, ganster] .

15 ?- noun_phrase(NP).
NP = [the, talented, writer] .

15 ?- sentence(S).
S = [the, beautiful, writer, challenged, the, yummy, writer] .

16 ?- sentence(S).
S = [the, ridiculiouus, player, fought, the, yummy, dogwalker] .

16 ?- sentence(S).
S = [the, funny, babysitter, ran, the, based, writer] .

16 ?-
sentence(S).
S = [the, ridiculiouus, babysitter, challenged, the, funny, gamer] .

16 ?- sentence(S).
S = [the, talented, artist, ran, the, yummy, programmer] .

16 ?- sentence(S).
S = [the, yummy, artist, dressed, the, beautiful, dogwalker] .

16 ?- sentence(S).
S = [the, yummy, dogwalker, challenged, the, yummy, artist] .
```

Nathaniel Wolf
Prolog Programming Assignment #1: Various Computations
4/6/2022
CSC 344



```
16 ?- sentence(S).
S = [the, ridiculiouus, gamer, adminsatred, the, yummy, ganster] .

16 ?- sentence(S).
S = [the, beautiful, programmer, dressed, the, talented, programmer] .

16 ?- sentence(S).
S = [the, beautiful, gamer, dressed, the, funny, ganster] .

16 ?- sentence(S).
S = [the, beautiful, babysitter, dressed, the, beautiful, dogwalker] .

16 ?- sentence(S).
S = [the, funny, writer, challenged, the, funny, artist] .

16 ?- sentence(S).
S = [the, beautiful, dogwalker, dressed, the, ridiculiouus, babysitter] .

16 ?- sentence(S).
S = [the, beautiful, programmer, dressed, the, beautiful, dogwalker] .

16 ?- sentence(S).
S = [the, based, artist, ran, the, talented, dogwalker] .

16 ?- sentence(S).
S = [the, talented, programmer, cut, the, talented, player] .

16 ?- sentence(S).
S = [the, yummy, player, challenged, the, talented, dogwalker] .

16 ?- sentence(S).
S = [the, talented, programmer, proctored, the, funny, artist] .
```