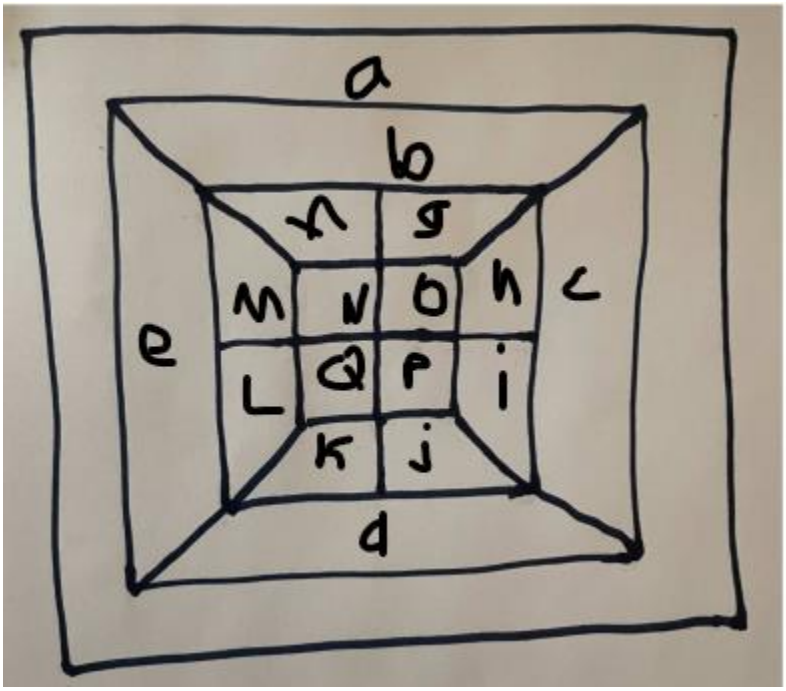
Prolog Programming Assignment #1: Various Computations

Learning Abstract

This Prolog Programming Assignment #1 is a total of 4 tasks. In task 1 we create a knowledge base in order to color a map using 4 different colors. Task 2 introduces predicate design in which we follow along with a demo to learn thru interaction. In task 3 involves Pokémon in which we copy a knowledge base and extend it to produce a demo. Task 4 involves list processing using different demos from lesson 5. Overall the assignment is excellent practice and introduction to the Prolog Language.

Task 1: Map Coloring

Map:



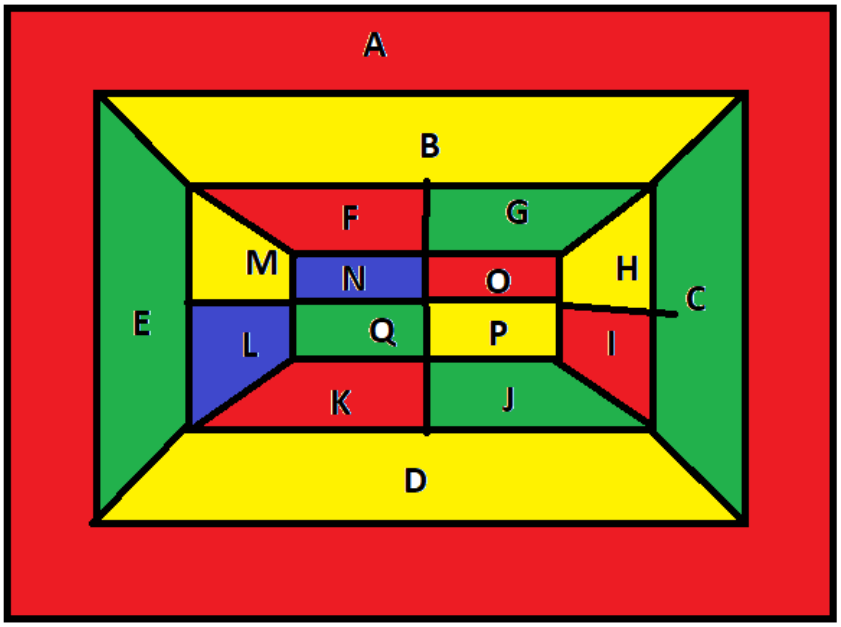
Code:

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

coloring(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q) :-
    different(A,B),
    different(A,C),
    different(A,D),
    different(A,E),
    different(B,C),
    different(B,E),
    different(B,F),
    different(B,G),
    different(C,D),
    different(C,H),
    different(C,I),
    different(D,E),
    different(D,K),
    different(D,J),
    different(E,M),
    different(E,L),
    different(F,M),
    different(F,G),
    different(F,N),
    different(G,O),
    different(G,N),
    different(H,O),
    different(H,I),
    different(I,P),
    different(I,J),
    different(J,P),
    different(J,K),
    different(K,L),
    different(K,Q),
    different(L,M),
    different(M,N),
    different(N,O),
    different(N,Q),
    different(O,P),
    different(P,Q).
```

Demo:

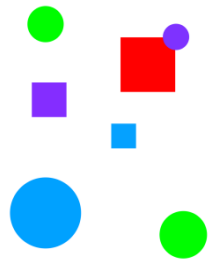
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\zchbo\Desktop\344 Programming Languages\Prolog> swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.2)
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('task1.pro').
true.

2 ?- coloring(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q).
A = F, F = I, I = K, K = O, O = red,
B = D, D = H, H = M, M = P, P = yellow,
C = E, E = G, G = J, J = Q, Q = green,
L = N, N = blue
```

Colored Map:



Task 2: The Floating Shapes World

Image:



Code:

```
% ---------------------------------------------------------------------
% ---------------------------------------------------------------------
% --- File: task2.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),nsl,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.
```

Demo:

```
1 ?- consult('task2.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 ?- circles.
carla
shapes :-
    circles,
    squares.

true.
```

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

8 ?- blue(shape).
false.

9 ?- blue(Shape).
Shape = sara .

10 ?- blue(Shape).
Shape = sara .

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

10 ?- large(Name),write(Name),nl,fail.
sera
sara
false.

12 ?- area(cora,A).
A = 153.86
Unknown action: A (h for help)
14 ?- halt.
```

Task 3: Pokemon KB Interaction and Programming

Part 1 Code:

```
% ------------------------------------------------------------------------
% ------------------------------------------------------------------------
% --- 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)).

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)).
```

Part 1 Demo:

```
For built-in help, use ?- help(Topic). or ?- apropos(Word).

1 ?- consult('task3.pro').
true.

2 ?- cen(pikachu).
true.

3 ?- cen(raichu).
false.

4 ?- cen(Name).
Name = pikachu ;
Name = bulbasaur .

5 ?- cen(Name).
Name = pikachu ;
Name = bulbasaur ;
Name = caterpie ;
Name = charmander ;
Name = vulpix ;
Name = poliwag ;
Name = squirtle ;
Name = staryu.

5 ?- cen(Name),write(Name),nl,fail.
pikachu
bulbasaur
caterpie
charmander
vulpix
poliwag
squirtle
staryu
false.

6 ?- evolves(squirtle,wartortle).
true.

7 ?- evolves(wartortle,squirtle).
false.
```

```
8 ?- evolves(squirtle,blastoise).
false.

9 ?- evolves(X,Y),evolves(Y,Z).
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.

10 ?- evolves(X,Y),evolves(Y,Z),write(X),write('-->'),wr
ite(Z),nl,fail.
bulbasaur-->venusaur
caterpie-->butterfree
charmander-->charizard
poliwag-->poliwrath
squirtle-->blastoise
false.
```

```
11 ?- pokemon(name(N),_,_,_),write(N),nl,fail.
pikachu
raichu
bulbasaur
ivysaur
venusaur
caterpie
metapod
butterfree
charmander
charmeleon
charizard
vulpix
ninetails
poliwag
poliwhirl
poliwrath
squirtle
wartortle
blastoise
staryu
starmie
false.

12 ?- pokemon(name(N),fire,_,_),write(N),nl,fail.
charmander
charmeleon
charizard
vulpix
ninetails
false.
```

```
13 ?- pokemon(name(N),T,_,_),write('nks(name('),write(N)
,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(poliwrath),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.

14 ?- pokemon(name(N),_,_,attack(waterfall,_)).
N = wartortle ;
false.

15 ?- pokemon(name(N),_,_,attack(poison-powder,_)).
N = venusaur
Unknown action: , (h for help)
waterfall
hydro-pump
slap
star-freeze
false.
```

```
17 ?- pokemon(name,poliwhirl),_,hp(HP),_).
ERROR: Syntax error: Illegal start of term
ERROR: pokemon(name,poliwhirl),_,hp(HP),
ERROR: ** here **
ERROR: _) .
17 ?- pokemon(name(poliwhirl),_,hp(HP),_).
HP = 80.

18 ?- pokemon(name(butterfree),_,hp(HP),_).
HP = 130.

19 ?- pokemon(name(N),_,hp(HP),_),HP>85,write(N),nl,fail.
raichu
poliwag: 60
squirtle: 40
staryu: 40
false.

23 ?-
```

Part 2 Code:

```
% ------------------------------------------------------------------------
% ------------------------------------------------------------------------
% --- 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)).

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)).

```

```
70
71  display_names :-
72      pokemon(name(N),_,_,_),write(N),nl,fail.
73
74  display_attacks :-
75      pokemon(_,_,_,attack(N,_)),write(N),nl,fail.
76
77  powerful(N) :-
78      pokemon(name(N),_,_,attack(_,Damage)),Damage>55.
79
80  tough(N) :-
81      pokemon(name(N),_,hp(HP),_),HP>100.
82
83  type(N,T) :-
84      pokemon(name(N),T,_,_).
85
86  dump_kind(T) :-
87      pokemon(name(N),T,hp(HP),attack(Attack, Damage)),
88      write('pokemon(name('),write(N),write('),'),write(T),write(', hp('),write(HP),
89      write('),attack('),write(Attack),write(', '),write(Damage),write('))).'),nl,fail.
90
91  display_cen :-
92      cen(Pokemon),write(Pokemon),nl,fail.
93
94  family(X) :-
95      evolves(X,Y),evolves(Y,Z),write(X),write(' '),write(Y),
96      write(' '),write(Z).
97
98  family(X) :-
99      evolves(X,Y),\+evolves(Y,_),write(X),write(' '),write(Y).
100
101 families :-
102     cen(N),family(N),nl,fail.
103
104 display_info(N) :-
105     pokemon(name(N),T,hp(HP),attack(A,D)),write('pokemon(name('),write(N),
106     write('),'),write(T),write(',hp('),write(HP),write('),attack('),write(A),
107     write(','),write(D),write(')).').
108
109 lineage(N) :-
110     evolves(N,O),evolves(O,P),display_info(N),nl,display_info(O),nl,display_info(P).
111
112 lineage(N) :-
113     evolves(N,O),display_info(N),nl,display_info(O).
114
115 lineage(N) :-
116     display_info(N).
```

Part 2 Demo:

```
1 ?- consult('C:/Users/zchbo/Desktop/344 Programming Languages/Prolog/task3.pro').
true.

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

```
3 ?- display_attacks.
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.

4 ?- powerful(pikachu).
false.

5 ?- powerful(blastoise).
true .

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

7 ?- tough(raichu).
false.

8 ?- tough(venusaur).
true.
```

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

10 ?- type(caterpie,grass).
true .

11 ?- type(pikachu,water).
false.

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

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

14 ?- 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.
```

```
15 ?- 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.

16 ?- display_cen.
pikachu
```

Task 3: Pokemon KB Interaction and Programming

Demo for Head/Tail Referencing Exercises:

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

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

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

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

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

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

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

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

Code:

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

rest([_|T],T).

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

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

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

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

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

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

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

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

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

make_set([H|T],[H|TS]) :-
    make_set(T,TS).
```

```
40
41  product([],1).
42  product([H|T],P) :-
43      product(T,PofT),P is H * PofT.
44
45  factorial(N,F) :-
46      iota(N,I),product(I,F).
47
48  make_list(0,_,_).
49  make_list(I,D,L) :-
50      J is I-1,make_list(J,D,R),add_last(D,R,L).
51
52  but_first(N,L) :-
53      rest(N,L).
54
55  but_last([],[]).
56  but_last([_|[]],[]).
57  but_last([H|T],L) :-
58      but_last(T,R),add_first(H,R,L).
59
60  is_palidrome([]).
61  is_palidrome([_|[]]).
62  is_palidrome(L) :-
63      first(L,F),last(L,Last), F=Last,but_first(L,R),but_last(R,S),is_palidrome(S).
64
65  noun_phrase(NP) :-
66      pick([happy,mad,sad,upset,lovingly,jealous],A),pick([car,house,man,bike,zebra,cow,truck,roof],N),add_last(A,[the],L),add_last(N,L,NP).
67
68  sentence(S) :-
69      pick([expunged,delted,burned,destroyed,bounced,eaten,spanked],V),noun_phrase(NP1),noun_phrase(NP2),add_last(V,NP1,N1),
70      nth(0,NP2,E1),nth(1,NP2,E2),nth(2,NP2,E3),add_last(E1,N1,N2),add_last(E2,N2,N3),add_last(E3,N3,S).
```

Demo for Example List Processors:

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

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

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

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

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

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

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

8 ?-  nth(0,[zero,one,two,three,four],Element).
Element = zero .

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

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

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

12 ?- sum([2,3,5,7,11],SumOfPrimes).
SumOfPrimes = 28.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Demo for List Processing Exercises:

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

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 .

6 ?-  make_list(7,seven,Seven).
Seven = [seven, seven, seven, seven, seven, seven, seven] .

7 ?- make_list(8,2,List).
List = [2, 2, 2, 2, 2, 2, 2, 2] .

8 ?- ?- but_first([a,b,c],X).
ERROR: Unknown procedure: (?-)/1
ERROR:   ?- is the Prolog prompt
ERROR:   See FAQ at https://www.swi-prolog.org/FAQ/ToplevelM
ERROR: In:
ERROR:    [9] throw(error(existence_error(procedure,...),_65
ERROR:    [6] correct_goal((?-but_first(...,_6566)),user,['X
ERROR:
ERROR: Note: some frames are missing due to last-call optimi
ERROR: Re-run your program in debug mode (:- debug.) to get
9 ?- but_first([a,b,c],X).
X = [b, c].

10 ?- but_last([a,b,c,d,e],X)

.
X = [a, b, c, d] .

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

11 ?- is_palindrome([x]).
Correct to: "is_palidrome([x])"?
Please answer 'y' or 'n'? yes
true .
```

```
12 ?-  is_palindrome([a,b,c])
.
Correct to: "is_palidrome([a,b,c])"? yes
false.

13 ?-
is_palindrome([a,b,b,a])..

.
ERROR: Syntax error: Operator expected
ERROR: is_palindrome([a,b,b,a]
ERROR: ** here **
ERROR: ).. .
13 ?- is_palindrome([a,b,b,a]).
Correct to: "is_palidrome([a,b,b,a])"? yes
true .

14 ?- is_palindrome([1,2,3,4,5,4,2,3,1]).
Correct to: "is_palidrome([1,2,3,4,5,4,2,3,1])"?
Please answer 'y' or 'n'? yes
false.

15 ?-
 is_palindrome([c,o,f,f,e,e,e,e,f,f,o,c]).
Correct to: "is_palidrome([c,o,f,f,e,e,e,e,f,f,o,c])"?
Please answer 'y' or 'n'? yes
true .

16 ?- noun_phrase(NP).
NP = [the, happy, roof] ;
false.

17 ?- noun_phrase(NP).
NP = [the, upset, roof] ;
false.

17 ?-
noun_phrase(NP).
NP = [the, upset, car] ;
false.

17 ?-
noun_phrase(NP).
NP = [the, upset, bike] .
```

```
17 ?- noun_phrase(NP).
NP = [the, lovingly, zebra] ;
false.

17 ?-  sentence(S).
S = [the, sad, man, eaten, the, sad, car] ;
false.

18 ?-  sentence(S).
S = [the, lovingly, man, bounced, the, lovingly, zebra] .

18 ?-  sentence(S).
S = [the, happy, bike, expunged, the, sad, roof] .

18 ?-  sentence(S).
S = [the, mad, man, bounced, the, lovingly, man] .

18 ?-  sentence(S).
S = [the, sad, car, spanked, the, sad, truck] .

18 ?-  sentence(S).
S = [the, upset, man, delted, the, lovingly, roof] .

18 ?-  sentence(S).
S = [the, lovingly, truck, spanked, the, happy, cow] .

18 ?-  sentence(S).
S = [the, upset, car, bounced, the, happy, truck] .

18 ?-  sentence(S).
S = [the, sad, man, burned, the, sad, man] .

18 ?-  sentence(S).
S = [the, mad, man, destroyed, the, upset, bike] .

18 ?-  sentence(S).
S = [the, sad, roof, eaten, the, upset, bike] .

18 ?-  sentence(S).
S = [the, mad, truck, delted, the, lovingly, bike] .

18 ?-
 sentence(S).
S = [the, sad, zebra, burned, the, happy, man] .
```