```
/* Prolog prototype for a generalized N-Queens puzzle solver.
   Spatial positioning is conveyed as X/Y, where X ~ col and Y ~ row.

   Michael E Sparks, 30 Oct 2020
*/

% This predicate tests whether the queen at position
% (X1,Y1) is safe w/r/t all other placed queens.
safe(_,[]).
safe(X1/Y1,[X2/Y2|Rest]) :-
    X1 =\= X2, % column is safe
    Y1 =\= Y2, % row is safe
    abs(X1 - X2) =\= abs(Y1 - Y2), % diagonals are safe
    safe(X1/Y1,Rest).

/*
?- safe(1/2,[2/4,3/1,4/3]).
true ;
false.

?- safe(1/2,[3/1,4/3]).
true ;
false.

?- safe(1/2,[4/3]).
true ;
false.

?- safe(1/2,[]).
true.
*/

% As is, proper use of the following predicate would
% require masking either the X's or the Y's.
soln1(_,[]).
soln1(N,[X/Y|Rest]) :-
    soln1(N,Rest),
    between(1,N,X),
    between(1,N,Y),
    safe(X/Y,Rest).

/*
?- soln1(4,[X1/Y1,X2/Y2,X3/Y3,X4/Y4]), X1 = 1, X2 = 2, X3 = 3, X4 = 4.
X1 = Y2, Y2 = 1,
Y1 = X3, X3 = 3,
X2 = Y4, Y4 = 2,
Y3 = X4, X4 = 4 ;
X1 = Y3, Y3 = 1,
Y1 = X2, X2 = 2,
Y2 = X4, X4 = 4,
X3 = Y4, Y4 = 3 ;
false.

?- soln1(4,[1/Y1,2/Y2,3/Y3,4/Y4]).
Y1 = 3,
Y2 = 1,
Y3 = 4,
Y4 = 2 ;
Y1 = 2,
Y2 = 4,
Y3 = 1,
Y4 = 3 ;
false.
*/

% This fully generalized predicate coerces the solution
% S to be of length N, and calls a helper predicate that
% tracks its depth in the recursion.
soln(N,S) :-
```

```
    length(S,N),
    soln_aux(N,N,S).

soln_aux(_,_,[]).
soln_aux(N,D,[X/Y|Rest]) :-
    D1 is D - 1,
    soln_aux(N,D1,Rest),
    X = D,
    between(1,N,Y),
    safe(X/Y,Rest).

/*
?- soln(4,S).
S = [4/3, 3/1, 2/4, 1/2] ;
S = [4/2, 3/4, 2/1, 1/3] ;
false.

?- bagof(S,soln(4,S),Sols), length(Sols,Num_sols).
Sols = [[4/3, 3/1, 2/4, 1/2], [4/2, 3/4, 2/1, 1/3]],
Num_sols = 2.

?- soln(5,S).
S = [5/4, 4/2, 3/5, 2/3, 1/1] ;
S = [5/3, 4/5, 3/2, 2/4, 1/1] ;
S = [5/5, 4/3, 3/1, 2/4, 1/2] ;
S = [5/4, 4/1, 3/3, 2/5, 1/2] ;
S = [5/5, 4/2, 3/4, 2/1, 1/3] ;
S = [5/1, 4/4, 3/2, 2/5, 1/3] ;
S = [5/2, 4/5, 3/3, 2/1, 1/4] ;
S = [5/1, 4/3, 3/5, 2/2, 1/4] ;
S = [5/3, 4/1, 3/4, 2/2, 1/5] ;
S = [5/2, 4/4, 3/1, 2/3, 1/5] ;
false.

?- bagof(S,soln(5,S),Sols), length(Sols,Num_sols).
Sols = [[5/4, 4/2, 3/5, 2/3, 1/1], [5/3, 4/5, 3/2, 2/4, 1/1], [5/5, 4/3, 3/1, 2/
4, 1/2], [5/4, 4/1, 3/3, 2/5, ... / ...], [5/5, 4/2, 3/4, ... / ...|...], [5/1,
4/4, ... / ...|...], [5/2, ... / ...|...], [... / ...|...], [...|...]|...],
Num_sols = 10.

?- bagof(S,soln(8,S),Sols), length(Sols,Num_sols).
Sols = [[8/4, 7/2, 6/7, 5/3, 4/6, 3/8, 2/5, ... / ...], [8/5, 7/2, 6/4, 5/7, 4/3
, 3/8, ... / ...|...], [8/3, 7/5, 6/2, 5/8, 4/6, ... / ...|...], [8/3, 7/6, 6/4,
 5/2, ... / ...|...], [8/5, 7/7, 6/1, ... / ...|...], [8/4, 7/6, ... / ...|...],
 [8/3, ... / ...|...], [... / ...|...], [...|...]|...],
Num_sols = 92.

?- bagof(S,soln(6,S),Sols), length(Sols,Num_sols).
Sols = [[6/5, 5/3, 4/1, 3/6, 2/4, 1/2], [6/4, 5/1, 4/5, 3/2, 2/6, 1/3], [6/3, 5/
6, 4/2, 3/5, 2/1, ... / ...], [6/2, 5/4, 4/6, 3/1, ... / ...|...]],
Num_sols = 4.

?- use_module(library(statistics)).
true.

?- time((bagof(S,soln(13,S),Sols), length(Sols,Num_sols))).
% 1,002,450,941 inferences, 157.889 CPU in 157.896 seconds (100% CPU, 6349104 Li
ps)
Sols = [[13/7, 12/11, 11/8, 10/6, 9/4, 8/13, 7/10, ... / ...|...], [13/12, 12/10
, 11/8, 10/6, 9/4, 8/2, ... / ...|...], [13/9, 12/11, 11/8, 10/2, 9/4, ... / ...
|...], [13/11, 12/8, 11/6, 10/2, ... / ...|...], [13/7, 12/11, 11/6, ... / ...|.
..], [13/12, 12/7, ... / ...|...], [13/9, ... / ...|...], [... / ...|...], [...|
...]|...],
Num_sols = 73712.
*/
```