```
% Kolorowanie mapy ------------------------------------------------------------------------------------------
% A, B, C, D, E to pola mapy, a pozniej ograniczenia, ktore nie moze byc przy ktorym
solutionMC(A,B,C,D,E) :-
  color(A), color(B), color(C), color(D), color(E),
  \+ A=B, \+ A=C, \+ A=D, \+ A=E, \+ B=C, \+ C=D, \+ D=E.

color(red).
color(white).
color(blue).



% Sudoku ------------------------------------------------------------------------------------------

 Problem rozwiązywany jest przez predykat sudoku z następującego pliku: sudoku.pl.
 Predykat uniq jest jednak niepoprawnie napisany – powinien on sprawdzać,
 czy zmienne P, Q, R, S są różne. Przyjrzyj się jednak predykatowi solution z tego pliku.
 Używa on predykatu uniq nie tylko do testowania, ale także do generowania rozwiązań.
 Napisz ten predykat tak, aby unifikował zmienne z odpowiednimi liczbami w razie potrzeby.

Z poprawnym predykatem uniq predykat solution będzie mógł sprawdzać,
czy w poszczególnych wierszach, kolumnach i kwadratach 2×2 są różne liczby.
Przetestuj działanie programu wywołując następujący cel:

sudoku(
1, 4, _, _,
_, _, 4, _,
2, _, _, _,
_, _, _, 3).


% The main predicate. Solve the puzzle and print the answer.
% The variable Rij stands for the number in row i and column j.
sudoku(R11,R12,R13,R14,R21,R22,R23,R24,R31,R32,R33,R34,
         R41,R42,R43,R44) :-
   solution(R11,R12,R13,R14,R21,R22,R23,R24,R31,R32,R33,R34,
              R41,R42,R43,R44),
   nl, write('A solution to this puzzle is'), nl,
   printrow(R11,R12,R13,R14), printrow(R21,R22,R23,R24),
   printrow(R31,R32,R33,R34), printrow(R41,R42,R43,R44).

% Print a row of four numbers with spaces between them.
printrow(P,Q,R,S) :- write('  '), write(P), write(' '), write(Q),
   write(' '), write(R), write(' '), write(S), nl.

%-------------------------------------------------------------------
solution(R11,R12,R13,R14,R21,R22,R23,R24,R31,R32,R33,R34,
           R41,R42,R43,R44) :-
   uniq(R11,R12,R13,R14), uniq(R21,R22,R23,R24),       % rows 1,2
   uniq(R31,R32,R33,R34), uniq(R41,R42,R43,R44),       % rows 3,4
   uniq(R11,R21,R31,R41), uniq(R12,R22,R32,R42),       % cols 1,2
   uniq(R13,R23,R33,R43), uniq(R14,R24,R34,R44),       % cols 3,4
   uniq(R11,R12,R21,R22), uniq(R13,R14,R23,R24),       % NW and NE
   uniq(R31,R32,R41,R42), uniq(R33,R34,R43,R44).       % SW and SE

% uniq holds if P,Q,R,S are all distinct nums (from 1 to 4).
uniq(P,Q,R,S) :- fail.

% The four numbers to go into each cell
num(1).  num(2).  num(3).  num(4).



% Kryptografia ------------------------------------------------------------------------------------------

Problemy kryptoarytmetyczne są klasą zagadek, w których musimy odgadnąć,
jakim cyfrom odpowiadają litery
umieszczone w pewnym równaniu. Klasycznym przykładem takiego problemu
jest SEND+MORE=MONEY:

  SEND
 +MORE
------
 MONEY

Za poszczególne litery (S,E,N,D,M,O,R,Y) musimy wstawić różne cyfry od 0 do 9 tak,
aby przedstawione dodawanie było poprawne. Zmiennym S i M
muszą odpowiadać przy tym wartości różne od zera.



% solution(...) holds for a solution to SEND+MORE=MONEY.
solution(S,E,N,D,M,O,R,Y) :-
   uniq_digits(S,E,N,D,M,O,R,Y), S > 0, M > 0,
   Y is (D+E) mod 10, C1 is (D+E) // 10,
   E is (N+R+C1) mod 10, C10 is (N+R+C1) // 10,
   N is (E+O+C10) mod 10, C100 is (E+O+C10) // 10,
   O is (S+M+C100) mod 10, M is (S+M+C100) // 10.

% uniq(...) holds if the arguments are all distinct digits.
uniq_digits(S,E,N,D,M,O,R,Y) :-
```

```
   dig(S), dig(E), dig(N), dig(D), dig(M), dig(O), dig(R), dig(Y),
   \+ S=E, \+ S=N, \+ S=D, \+ S=M, \+ S=O, \+ S=R, \+ S=Y,
           \+ E=N, \+ E=D, \+ E=M, \+ E=O, \+ E=R, \+ E=Y,
                   \+ N=D, \+ N=M, \+ N=O, \+ N=R, \+ N=Y,
                           \+ D=M, \+ D=O, \+ D=R, \+ D=Y,
                                   \+ M=O, \+ M=R, \+ M=Y,
                                           \+ O=R, \+ O=Y,
                                                   \+ R=Y.
% The digits
dig(0). dig(1). dig(2). dig(3). dig(4).
dig(5). dig(6). dig(7). dig(8). dig(9).


% Problem ośmiu hetmanów --------------------------------------------------------------------------------------
```

Problem ośmiu hetmanów polega na rozmieszczeniu na szachownicy 8×8 ośmiu hetmanów (królowych) tak,
aby żadna nie biła dowolnej innej. Aby to zachodziło, wszystkie muszą być w różnych wierszach,
kolumnach, lewych oraz prawych przekątnych.

Do opisania tego problemu użyjemy ośmiu zmiennych: C1, …, C8, gdzie Ci oznacza kolumnę w której
znajduje się hetman z i-tego rzędu. Ten sposób kodowania zapewnia, że żadni dwaj hetmani nie
znajdę się w tym samym wierszu. Pozostałe warunki trzeba jednak sprawdzać. Niezajmowanie tej
samej kolumny sprawdzić jest prosto. Można również wykazać, że dla pól na tej samej lewej
czy prawej przekątnej odpowiednio różnica i suma współrzędnych są stałe.

```
% Solve the 8-queens problem.
solution(C1,C2,C3,C4,C5,C6,C7,C8) :-
  col(C1),
  col(C2), \+ cap(2,C2,1,C1),
  col(C3), \+ cap(3,C3,1,C1), \+ cap(3,C3,2,C2),
  col(C4), \+ cap(4,C4,1,C1), \+ cap(4,C4,2,C2), \+ cap(4,C4,3,C3),
  col(C5), \+ cap(5,C5,1,C1), \+ cap(5,C5,2,C2), \+ cap(5,C5,3,C3),
      \+ cap(5,C5,4,C4),
  col(C6), \+ cap(6,C6,1,C1), \+ cap(6,C6,2,C2), \+ cap(6,C6,3,C3),
      \+ cap(6,C6,4,C4), \+ cap(6,C6,5,C5),
  col(C7), \+ cap(7,C7,1,C1), \+ cap(7,C7,2,C2), \+ cap(7,C7,3,C3),
      \+ cap(7,C7,4,C4), \+ cap(7,C7,5,C5), \+ cap(7,C7,6,C6),
  col(C8), \+ cap(8,C8,1,C1), \+ cap(8,C8,2,C2), \+ cap(8,C8,3,C3),
      \+ cap(8,C8,4,C4), \+ cap(8,C8,5,C5), \+ cap(8,C8,6,C6),
           \+ cap(8,C8,7,C7).

% The columns
col(1). col(2). col(3). col(4). col(5). col(6). col(7). col(8).

% cap(R1,C1,R2,C2): a queen on (R1,C1) can capture one on (R2,C2).
cap(R,_,R,_).                           % Note the use of the _ here
cap(_,C,_,C).                           % and here, too.
cap(R1,C1,R2,C2) :- R1-C1 =:= R2-C2.
cap(R1,C1,R2,C2) :- R1+C1 =:= R2+C2.
```