


Tower of Hanoi

```
move(1, X, Y, _) :-  
    write('Move top disk from '),  
    write(X),  
    write(' to '),  
    write(Y),  
    nl.
```

```
move(N, X, Y, Z) :-  
    N > 1,  
    M is N - 1,  
    move(M, X, Z, Y),  
    move(1, X, Y, _),  
    move(M, Z, Y, X).
```

```
hanoi(N) :-  
    move(N, left, right, middle).
```

Output:



```
hanoi(3).  
Move top disk from left to right  
Move top disk from left to middle  
Move top disk from right to middle  
Move top disk from left to right  
Move top disk from middle to left  
Move top disk from middle to right  
Move top disk from left to right  
true  
1  
Next 10 100 1,000 Stop  
?- hanoi(3).
```

N-Queen

queens(N, Queens) :-

```
length(Queens, N),  
  board(Queens, Board, 0, N, _, _),  
  queens(Board, 0, Queens).
```

board([], [], N, N, _, _).

board([_|Queens], [Col-Vars|Board], Col0, N, [_|VR], VC) :-

```
Col is Col0+1,  
  functor(Vars, f, N),  
  constraints(N, Vars, VR, VC),  
  board(Queens, Board, Col, N, VR, [_|VC]).
```

constraints(0, _, _, _) :- !.

constraints(N, Row, [R|Rs], [C|Cs]) :-

```
arg(N, Row, R-C),  
M is N-1,  
constraints(M, Row, Rs, Cs).
```

queens([], _, []).

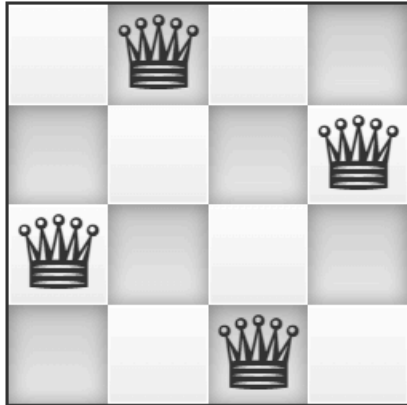
queens([C|Cs], Row0, [Col|Solution]) :-

```
Row is Row0+1,  
  select(Col-Vars, [C|Cs], Board),  
  arg(Row, Vars, Row-Row),  
  queens(Board, Row, Solution).
```

Output:

```
🛠️ queens(4, Queens).
```

Queens =



Queens =

