Q-8 Write a program to solve 8 puzzle problem using Prolog.

```
% 8-Puzzle Problem in Prolog

% Define possible moves (Left and Right)
move([X,0,Z,A,B,C,D,E,F],
     [0,X,Z,A,B,C,D,E,F]). % move left
move([X,Z,0,A,B,C,D,E,F],
     [X,0,Z,A,B,C,D,E,F]).
move([A,B,C,X,0,Z,D,E,F],
     [A,B,C,0,X,Z,D,E,F]).
move([A,B,C,X,Z,0,D,E,F],
     [A,B,C,X,0,Z,D,E,F]).
move([A,B,C,D,E,F,X,0,Z],
     [A,B,C,D,E,F,0,X,Z]).
move([A,B,C,D,E,F,X,Z,0],
     [A,B,C,D,E,F,X,0,Z]).

% Right moves

move([0,X,Z,A,B,C,D,E,F],
     [X,0,Z,A,B,C,D,E,F]).
move([X,0,Z,A,B,C,D,E,F],
     [X,Z,0,A,B,C,D,E,F]).
move([A,B,C,0,X,Z,D,E,F],
     [A,B,C,X,0,Z,D,E,F]).
move([A,B,C,X,0,Z,D,E,F],
     [A,B,C,X,Z,0,D,E,F]).
```

```
move(C[A,B,C,D,E,F,0,X,Z],
     [A,B,C,D,E,F,X,0,Z]).
move(C[A,B,C,D,E,F,X,0,Z],
     [A,B,C,D,E,F,X,Z,0]).

% Up moves
move(C[A,B,C,D,E,F,0,X,Z],
     [A,B,C,0,E,F,D,X,Z]).
move(C[A,B,C,D,E,F,X,0,Z],
     [A,B,C,D,0,F,X,E,Z]).
move(C[A,B,C,D,E,F,X,Z,0],
     [A,B,C,D,E,0,X,Z,F]).

% Down moves
move(C[0,B,C,A,E,F,D,H,I],
     [A,B,C,0,E,F,D,H,I]).
move(C[A,0,C,D,E,F,G,H,I],
     [A,E,C,D,0,F,G,H,I]).
move(C[A,B,0,D,E,F,G,H,I],
     [A,B,F,D,E,0,G,H,I]).

% Solve function using depth-first search
solve(State,Goal):-
    dfs([State],Goal,Path),
    write('Solution Path:'),nl,
    print_path(Path).
```

```
% Depth-First Search
dfs([Goal|Visited], Goal, [Goal|Visited]).
dfs([Current|Visited], Goal, Path) :-
    move(Current, Next),
    \+ member(Next, Visited),
    dfs([Next, Current|Visited], Goal, Path).

% Print solution path
print_path([]).
print_path([H|T]) :-
    print_state(H),
    nl,
    print_path(T).

% Display board state in 3x3 format
print_state([A, B, C, D, E, F, G, H, I]) :-
    format('~w ~w ~w~n~w ~w ~w~n~w ~w ~w~n',
           [A, B, C, D, E, F, G, H, I]).
```

* Output

Input : ?- solve([[1,2,3,4,0,5,6,7,8],
[1,2,3,4,5,0,6,7,8]]).

Output : Solution Path:

1 2 3
4 5
6 7 8
1 2 3
4 5
6 7 8