

AI PRACTICAL NO. 8

Name – Shreya Panikkassery

Class – TE Comps A

Roll no – 9562

1. Prolog for tower of Hanoi

% hanoi(+N, +A, +B, +C, -Moves)

% N: number of disks

% A: the name of the source peg

% B: the name of the auxiliary peg

% C: the name of the destination peg

% Moves: list of moves to solve the Tower of Hanoi problem

hanoi(0, _, _, _, []).

hanoi(N, A, B, C, Moves) :-

 N > 0,

 N1 is N - 1,

 hanoi(N1, A, C, B, Moves1),

 append(Moves1, [(A,C)], Moves2),

 hanoi(N1, B, A, C, Moves3),

 append(Moves2, Moves3, Moves).

% Example usage:

% ?- hanoi(3, left, middle, right, Moves).

% Moves = [(left, right), (left, middle), (right, middle), (left, right), (middle, left), (middle, right), (left, right)].

% ?- hanoi_moves(4, Count).

% Count = 15

```
% ?- between(1, 5, N), hanoi(N, left, middle, right, Moves), writeln(Moves).
```

```
% [(left,right)]
```

```
% Moves = [(left,right)],
```

```
% N = 1
```

% If you only want to count the number of moves without storing them:

```
% hanoi_moves(+N, -Count)
```

```
hanoi_moves(N, Count) :-
```

```
    hanoi(N, _, _, _, Moves),
```

```
    length(Moves, Count).
```

OUTPUT:

The screenshot shows the SWISH Prolog environment. The left pane contains the following Prolog code:

```
1 % hanoi(+N, +A, +B, +C, -Moves)
2 % N: number of disks
3 % A: the name of the source peg
4 % B: the name of the auxiliary peg
5 % C: the name of the destination peg
6 % Moves: List of moves to solve the Tower of Hanoi problem
7 hanoi(0, _, _, _, []).
8 hanoi(N, A, B, C, Moves) :-
9     N > 0,
10    N1 is N - 1,
11    hanoi(N1, A, C, B, Moves1),
12    append(Moves1, [(A,C)], Moves2),
13    hanoi(N1, B, A, C, Moves3),
14    append(Moves2, Moves3, Moves).
15
16 % Example usage:
17 ?- hanoi(3, left, middle, right, Moves).
18 % Moves = [(left, right), (left, middle), (right, middle), (left, right), (middle, left), (middle, right), (left, right)].
19
20 ?- hanoi_moves(4, Count).
21 % Count = 15
22
23 ?- between(1, 5, N), hanoi(N, left, middle, right, Moves), writeln(Moves).
24 % [(left,right)]
25 % Moves = [(left,right)],
26 % N = 1
27
```

The right pane shows the execution results for three queries:

- Query 1:** `hanoi(3, left, middle, right, Moves).`
Result: `Moves = [(left,right), (left,middle), (right,middle), (left,right), (middle,left), (middle,right), (left,right)]`
Status: `false`
- Query 2:** `hanoi_moves(4, Count).`
Result: `Count = 15`
Controls: Next, 10, 100, 1,000, Stop
- Query 3:** `between(1, 5, N), hanoi(N, left, middle, right, Moves), writeln(Moves).`
Result: `[(left,right)]`
Result: `Moves = [(left,right)].`
Result: `N = 1`
Controls: Next, 10, 100, 1,000, Stop

At the bottom, there is a query prompt: `?- between(1, 5, N), hanoi(N, left, middle, right, Moves), writeln(Moves).` and buttons for Examples, History, Solutions, and a Run! button.

2. Prolog for N- Queen

% render solutions nicely.

:- use_rendering(chess).

```
%%    n_queens(?N, ?Cols) is nondet.  
  
%  
  
%    @param The k-th element of Cols is the column number of the  
%    queen in row k.  
%    @author Markus Triska
```

```
:- use_module(library(clpfd)).
```

```
n_queens(N, Qs) :-  
    length(Qs, N),  
    Qs ins 1..N,  
    safe_queens(Qs).
```

```
safe_queens([]).
```

```
safe_queens([Q|Qs]) :-  
    safe_queens(Qs, Q, 1),  
    safe_queens(Qs).
```

```
safe_queens([], _, _).
```

```
safe_queens([Q|Qs], Q0, D0) :-  
    Q0 #\= Q,  
    abs(Q0 - Q) #\= D0,  
    D1 #= D0 + 1,  
    safe_queens(Qs, Q0, D1).
```

```
/** <examples>
```

?- n_queens(8, Qs), labeling([ff], Qs).

?- n_queens(24, Qs), labeling([ff], Qs).

?- n_queens(100, Qs), labeling([ff], Qs).

*/

The screenshot shows the SWISH Prolog environment in a Mozilla Firefox browser. The address bar displays the URL https://swish.swi-prolog.org/example/clpfd_queens.pl. The SWISH interface includes a menu bar (File, Edit, Examples, Help), a search bar, and a list of open files. The main editor displays the Prolog code for `n_queens.pl`, which defines a predicate `n_queens(N, Qs)` and a `safe_queens` helper. The code includes comments and example queries. The right-hand pane shows the execution of the query `?- n_queens(8, Qs), labeling([ff], Qs).`. It features a chessboard visualization with 8 queens placed on the board, a list of solutions (Qs), and a 'Run!' button. The bottom status bar indicates the user is logged in as 'SWISH - d...lla Firefox' and the time is 15:02.

```
10 :- use_module(library(clpfd)).
11
12 n_queens(N, Qs) :-
13     length(Qs, N),
14     Qs ins 1..N,
15     safe_queens(Qs).
16
17 safe_queens([]).
18 safe_queens([_|Qs]) :-
19     safe_queens(Qs, 0, 1),
20     safe_queens(Qs).
21
22 safe_queens([], _, _).
23 safe_queens([_|Qs], Q0, D0) :-
24     Q0 #= 0,
25     abs(Q0 - 0) #\= D0,
26     D1 #= D0 + 1,
27     safe_queens(Qs, Q0, D1).
28
29
30 /** <examples>
31
32 ?- n_queens(8, Qs), labeling([ff], Qs).
33 ?- n_queens(24, Qs), labeling([ff], Qs).
34 ?- n_queens(100, Qs), labeling([ff], Qs).
35
36 */
37
```