Prolog Assignment 2

Learning Abstract: This assignment consisted of solving the Towers of Hanoi state space problem. The tasks of the assignment took me through putting together a program that solves to problem for 3 and 4 disks.

Task 3:

```
m12([Tower1Before,Tower2Before,Tower3],[Tower1After,Tower2After,Towe
    Tower1Before = [H|T],
    Tower1After = T,
    Tower2Before = L,
    Tower2After = [H|L].

test__m12 :-
    write('Testing: move_m12\n'),
    TowersBefore = [[t,s,m,l,h],[],[]],
    trace('','TowersBefore',TowersBefore),
    m12(TowersBefore,TowersAfter),
    trace('','TowersAfter',TowersAfter).
```

```
?-
|    test__m12.
Testing: move_m12
TowersBefore = [[t,s,m,l,h],[],[]]
TowersAfter = [[s,m,l,h],[t],[]]
true.

?-
```

Task 4:

```
m12([Tower1Before,Tower2Before,Tower3],[Tower1After,Tower2After,Tower3]) :-
    Tower1Before = [H|T],
    Tower1After = T,S
    Tower2Before = L,
    Tower2After = [H|L].

test__m12 :-
    write('Testing: move_m12\n'),
    TowersBefore = [[t,s,m,l,h],[],[]],
    trace('','TowersBefore',TowersBefore),
    m12(TowersBefore,TowersAfter),
    trace('','TowersAfter',TowersAfter).

m13([Tower1Before,Tower2,Tower3Before],[Tower1After,Tower2,Tower3After]) :-
    Tower1Before = [H|T],
    Tower1After = T,
    Tower3Before = L,
    Tower3After = [H|L].

test__m13 :-
    write('Testing: move_m13\n'),
    TowersBefore = [[t,s,m,l,h],[],[]],
    trace('','TowersBefore',TowersBefore),
    m13(TowersBefore,TowersAfter),
    trace('','TowersAfter',TowersAfter).

m21([Tower1Before,Tower2Before,Tower3],[Tower1After,Tower2After,Tower3]) :-
    Tower2Before = [H|T],
    Tower2After = T,
    Tower1Before = L,
    Tower1After = [H|L].

test__m21 :-
    write('Testing: move_m21\n'),
    TowersBefore = [[],[t,s,m,l,h],[]],
    trace('','TowersBefore',TowersBefore),
    m21(TowersBefore,TowersAfter),
    trace('','TowersAfter',TowersAfter).

m23([Tower1,Tower2Before,Tower3Before],[Tower1,Tower2After,Tower3After]) :-
    Tower2Before = [H|T],
    Tower2After = T,
    Tower3Before = L,
    Tower3After = [H|L].

test__m23 :-
    write('Testing: move_m23\n'),
    TowersBefore = [[],[t,s,m,l,h],[]],
    trace('','TowersBefore',TowersBefore),
    m23(TowersBefore,TowersAfter),
    trace('','TowersAfter',TowersAfter).
```

```
m31([Tower1Before,Tower2,Tower3Before],[Tower1After,Tower2,Tower3After]) :-
    Tower3Before = [H|T],
    Tower3After = T,
    Tower1Before = L,
    Tower1After = [H|L].

test__m31 :-
    write('Testing: move_m31\n'),
    TowersBefore = [[],[],[t,s,m,l,h]],
    trace('','TowersBefore',TowersBefore),
    m31(TowersBefore,TowersAfter),
    trace('','TowersAfter',TowersAfter).

m32([Tower1,Tower2Before,Tower3Before],[Tower1,Tower2After,Tower3After]) :-
    Tower3Before = [H|T],
    Tower3After = T,
    Tower2Before = L,
    Tower2After = [H|L].

test__m32 :-
    write('Testing: move_m32\n'),
    TowersBefore = [[],[],[t,s,m,l,h]],
    trace('','TowersBefore',TowersBefore),
    m32(TowersBefore,TowersAfter),
    trace('','TowersAfter',TowersAfter).
```

```
?-
|    test__m12.
Testing: move_m12
TowersBefore = [[t,s,m,l,h],[],[]]
TowersAfter = [[s,m,l,h],[t],[]]
true.

?- test__m13.
Testing: move_m13
TowersBefore = [[t,s,m,l,h],[],[]]
TowersAfter = [[s,m,l,h],[],[t]]
true.

?- test__m21.
Testing: move_m21
TowersBefore = [[],[t,s,m,l,h],[]]
TowersAfter = [[t],[s,m,l,h],[]]
true.

?- test__m23.
Testing: move_m23
TowersBefore = [[],[t,s,m,l,h],[]]
TowersAfter = [[],[s,m,l,h],[t]]
true.

?- test__m31.
Testing: move_m31
TowersBefore = [[],[],[t,s,m,l,h]]
TowersAfter = [[t],[],[s,m,l,h]]
true.

?- test__m32.
Testing: move_m32
TowersBefore = [[],[],[t,s,m,l,h]]
TowersAfter = [[],[t],[s,m,l,h]]
true.

?-
```

Task 5:

```
valid_state([Tower1,Tower2,Tower3]) :-
    valid_tower(Tower1), valid_tower(Tower2), valid_tower(Tower3).

valid_tower([]).
valid_tower([t]).
valid_tower([s]).
valid_tower([m]).
valid_tower([l]).
valid_tower([h]).

valid_tower([t,s]).
valid_tower([t,m]).
valid_tower([t,l]).
valid_tower([t,h]).
valid_tower([s,m]).
valid_tower([s,l]).
valid_tower([s,h]).
valid_tower([m,l]).
valid_tower([m,h]).
valid_tower([l,h]).

valid_tower([t,s,m]).
valid_tower([s,m,l]).
valid_tower([s,m,h]).
valid_tower([m,l,h]).

valid_tower([t,s,m,l]).
valid_tower([t,s,m,h]).
valid_tower([s,m,l,h]).

valid_tower([t,s,m,l,h]).
```

```
test__valid_state :-
  write('Testing:  valid_state\n'),
  test__vs([[l,t,s,m,h],[],[]]),
  test__vs([[t,s,m,l,h],[],[]]),
  test__vs([[],[h,t,s,m],[l]]),
  test__vs([[],[t,s,m,h],[l]]),
  test__vs([[],[h],[l,m,s,t]]),
  test__vs([[],[h],[t,s,m,l]]).

test__vs(S) :-
  valid_state(S),
  write(S), write(' is valid.'), nl.
test__vs(S) :-
  write(S), write(' is invalid.'), nl.
```

```
?- test__valid_state.
Testing: valid_state
[[l,t,s,m,h],[],[]] is invalid.
[[t,s,m,l,h],[],[]] is valid.
[[],[h,t,s,m],[l]] is invalid.
[[],[t,s,m,h],[l]] is valid.
[[],[h],[l,m,s,t]] is invalid.
[[],[h],[t,s,m,l]] is valid.
true
```

Task 6:

```
write_sequence([]).
write_sequence([H|T]) :-
    elaborate(H,E),
    write(E),nl,
    write_sequence(T).

elaborate(m12,Elaboration) :-
    Elaboration = 'Transfer a disk from tower 1 to tower 2.'.
elaborate(m13,Elaboration) :-
    Elaboration = 'Transfer a disk from tower 1 to tower 3.'.
elaborate(m21,Elaboration) :-
    Elaboration = 'Transfer a disk from tower 2 to tower 1.'.
elaborate(m23,Elaboration) :-
    Elaboration = 'Transfer a disk from tower 2 to tower 3.'.
elaborate(m31,Elaboration) :-
    Elaboration = 'Transfer a disk from tower 3 to tower 1.'.
elaborate(m32,Elaboration) :-
    Elaboration = 'Transfer a disk from tower 3 to tower 2.'.

test__write_sequence :-
    write('First test of write_sequence ...'), nl,
    write_sequence([m31,m12,m13,m21]),
    write('Second test of write_sequence ...'), nl,
    write_sequence([m13,m12,m32,m13,m21,m23,m13]).
```

```
?- test__write_sequence.
First test of write_sequence ...
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 1.
Second test of write_sequence ...
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 3 to tower 2.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 2 to tower 3.
Transfer a disk from tower 1 to tower 3.
true.

?-
```

Task 7:

```
?- solve.
PathSoFar = [[[s,m,l],[],[]]]
Move = m12
NextState = [[m,l],[s],[]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]]]
Move = m12
NextState = [[l],[m,s],[]]
Move = m13
NextState = [[l],[s],[m]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]]]
Move = m12
NextState = [[],[l,s],[m]]
Move = m13
NextState = [[],[s],[l,m]]
Move = m21
NextState = [[s,l],[],[m]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]]]
Move = m12
NextState = [[l],[s],[m]]
Move = m13
NextState = [[l],[],[s,m]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]]]
Move = m12
NextState = [[],[l],[s,m]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]]]
Move = m21
NextState = [[l],[],[s,m]]
Move = m23
NextState = [[],[],[l,s,m]]
Move = m31
NextState = [[s],[l],[m]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]]]
Move = m12
NextState = [[],[s,l],[m]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]]]
Move = m21
NextState = [[s],[l],[m]]
Move = m23
NextState = [[],[l],[s,m]]
Move = m31
NextState = [[m],[s,l],[]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]]]
Move = m12
NextState = [[],[m,s,l],[]]
Move = m13
NextState = [[],[s,l],[m]]
Move = m21
NextState = [[s,m],[l],[]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]]]
Move = m12
NextState = [[m],[s,l],[]]
Move = m13
NextState = [[m],[l],[s]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[m],[l],[s]]]
Move = m21
NextState = [[m],[l],[s]]
Move = m23
NextState = [[],[l],[m,s]]
Move = m31
NextState = [[s],[m,l],[]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[m],[l],[s]],[[],[m,l],[s]]]
Move = m12
NextState = [[],[s,m,l],[]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[m],[l],[s]],[[],[m,l],[s]],[[s],[m,l],[]]]
Move = m21
NextState = [[s],[m,l],[]]
Move = m23
NextState = [[],[m,l],[s]]
Move = m13
NextState = [[],[m,l],[s]]
Move = m21
NextState = [[m,s],[l],[]]
--

Move = m23
NextState = [[s],[l],[m]]
Move = m32
NextState = [[],[s,m,l],[]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[m],[l],[s]],[[],[m,l],[s]],[[],[s,m,l],[]]]
Move = m21
NextState = [[s],[m,l],[]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[m],[l],[s]],[[],[m,l],[s]],[[],[s,m,l],[]],[[s],[m,l],[]]]
Move = m12
NextState = [[],[s,m,l],[]]
Move = m13
NextState = [[],[m,l],[s]]
Move = m21
NextState = [[a,s],[l],[]]
Move = m23
NextState = [[s],[l],[m]]
Move = m23
NextState = [[],[m,l],[s]]
Move = m13
NextState = [[],[l],[m,s]]
Move = m21
NextState = [[l,m],[],[s]]
Move = m23
NextState = [[m],[],[l,s]]
Move = m31
NextState = [[s,m],[l],[]]
Move = m32
NextState = [[m],[s,l],[]]
Move = m21
NextState = [[l,s,m],[],[]]
Move = m23
NextState = [[s,m],[],[l]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[s,m],[],[l]]]
Move = m12
NextState = [[m],[s],[l]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[s,m],[],[l]],[[m],[s],[l]]]
Move = m12
NextState = [[],[m,s],[l]]
Move = m13
NextState = [[],[s],[l,m]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[s,m],[],[l]],[[m],[s],[l]],[[],[s],[m,l]]]
Move = m21
NextState = [[s],[],[m,l]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[s,m],[],[l]],[[m],[s],[l]],[[],[s],[m,l]],[[s],[m,l],[]]]
Move = m12
NextState = [[],[s],[m,l]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[s,m],[],[l]],[[m],[s],[l]],[[],[s],[m,l]],[[s],[m,l],[]],[[],[s,m,l],[]]]
Move = m21
NextState = [[s],[m,l],[]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[s,m],[],[l]],[[m],[s],[l]],[[],[s],[m,l]],[[s],[m,l],[]],[[],[s,m,l],[]],[[],[s,m,l]]]
Move = m21
NextState = [[s],[m,l],[]]
Move = m23
NextState = [[],[m,l],[s]]
Move = m13
NextState = [[],[m,l],[s]]
Move = m21
NextState = [[m,s],[l],[]]
--

Move = m23
NextState = [[s],[l],[m]]
Move = m32
NextState = [[],[s,m,l],[]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[m],[l],[s]],[[],[m,l],[s]],[[],[s,m,l],[]],[[s],[m,l],[]]]
Move = m12
NextState = [[m],[s],[l]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[s,m],[],[l]],[[m],[s],[l]],[[],[s],[m,l]],[[s],[m,l],[]],[[],[s,m,l],[]],[[],[s,m,l]]]
Move = m12
NextState = [[],[m,s],[l]]
Move = m13
NextState = [[],[s],[l,m]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[s,m],[],[l]],[[m],[s],[l]],[[],[s],[m,l]],[[s],[m,l],[]],[[],[s,m,l],[]],[[],[s,m,l]],[[s],[],[m,l]]]
Move = m21
NextState = [[s],[],[m,l]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[s,m],[],[l]],[[m],[s],[l]],[[],[s],[m,l]],[[s],[m,l],[]],[[],[s,m,l],[]],[[],[s,m,l]],[[s],[],[m,l]]]
Move = m12
NextState = [[],[s],[m,l]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[s,m],[],[l]],[[m],[s],[l]],[[],[s],[m,l]],[[s],[m,l],[]],[[],[s,m,l],[]],[[],[s,m,l]],[[s],[],[m,l]],[[],[s],[m,l]]]
Move = m13
NextState = [[],[],[s,m,l]]
PathSoFar = [[[s,m,l],[],[]],[[m,l],[s],[]],[[l],[s],[m]],[[s,l],[],[m]],[[l],[],[s,m]],[[],[l],[s,m]],[[s],[l],[m]],[[],[s,l],[m]],[[m],[s,l],[]],[[s,m],[l],[]],[[s,m],[],[l]],[[m],[s],[l]],[[],[s],[m,l]],[[s],[m,l],[]],[[],[s,m,l],[]],[[],[s,m,l]],[[s],[],[m,l]],[[],[s],[m,l]],[[],[],[s,m,l]]]
SolutionSoFar = [m12,m13,m21,m13,m12,m31,m12,m31,m21,m23,m12,m13,m21,m13]
```

```
Solution ...

Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 3.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 2 to tower 3.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 2 to tower 3.

true
```

(1) The length of my program solution is 14.
(2) The length of the shortest solution is 7.
(3) The program does not test every possible outcome, so it does not always choose the shortest path.

Task 8:

```
Solution ...

Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 3.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 3 to tower 1.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 1 to tower 3.
Transfer a disk from tower 2 to tower 1.
Transfer a disk from tower 2 to tower 3.
Transfer a disk from tower 1 to tower 2.
Transfer a disk from tower 2 to tower 3.

true
```

(1) The length is 35

(2) The shortest path is 17.