Tower of Hanoi

Tower of Hanoi is a mathematical puzzle where we have three rods and n disks. The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules:

1. Only one disk can be moved at a time.
2. Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.
3. No disk may be placed on top of a smaller disk.

Approach :

```
Let rod 1 = 'A', rod 2 = 'B', rod 3 = 'C'.
An example with2 disks :
Step 1 : Shift first disk from 'A' to 'B'.

Step 2 : Shift second disk from 'A' to 'C'.

Step 3 : Shift first disk from 'B' to 'C'.
```

```
An example with3 disks :
Step 1 : Shift first disk from 'A' to 'C'.
Step 2 : Shift second disk from 'A' to 'B'.
Step 3 : Shift first disk from 'C' to 'B'.

Step 4 : Shift third disk from 'A' to 'C'.

Step 5 : Shift first disk from 'B' to 'A'.
Step 6 : Shift second disk from 'B' to 'C'.
Step 7 : Shift first disk from 'A' to 'C'.
(Notice the gaps)
```

```
The pattern here is :
 - Shift 'n-1' disks from 'A' to 'B', using C.
```

```
  - Shift last disk from 'A' to 'C'.
  - Shift 'n-1' disks from 'B' to 'C', using A.
```

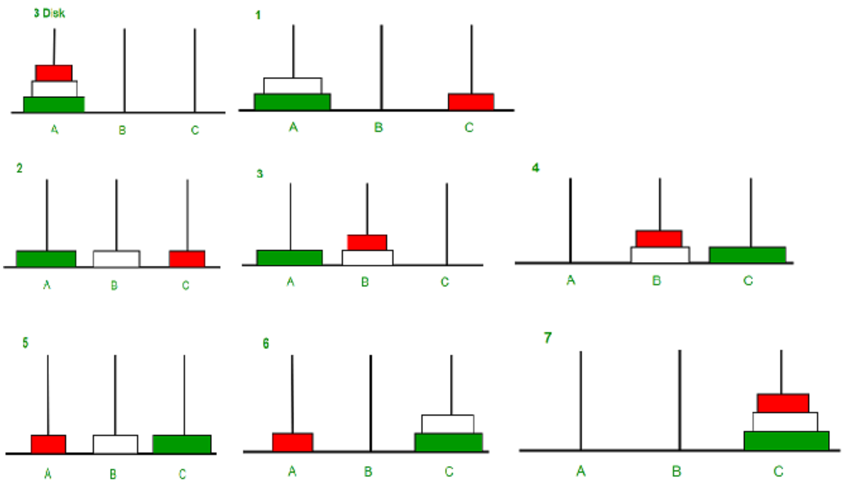

Image illustration for 3 disks

Examples:

```
Input : 2
Output : Disk 1 moved from A to B
         Disk 2 moved from A to C
         Disk 1 moved from B to C

Input : 3
Output : Disk 1 moved from A to C
         Disk 2 moved from A to B
         Disk 1 moved from C to B
         Disk 3 moved from A to C
         Disk 1 moved from B to A
         Disk 2 moved from B to C
         Disk 1 moved from A to C
```

```
// C++ recursive function to
// solve tower of hanoi puzzle
#include <bits/stdc++.h>
using namespace std;

void towerOfHanoi(int n, char from_rod,char to_rod, char aux_
{
    if (n == 0)
    {
        return;
    }
    towerOfHanoi(n - 1, from_rod, aux_rod, to_rod);
    cout << "Move disk " << n << " from rod " << from_rod <<
                                " to rod " << to_rod << endl;
    towerOfHanoi(n - 1, aux_rod, to_rod, from_rod);
}

// Driver code
int main()
{
    int n = 4; // Number of disks
    towerOfHanoi(n, 'A', 'C', 'B'); // A, B and C are names o
    return 0;
}
```

Output

```
Move disk 1 from rod A to rod B
Move disk 2 from rod A to rod C
Move disk 1 from rod B to rod C
Move disk 3 from rod A to rod B
Move disk 1 from rod C to rod A
Move disk 2 from rod C to rod B
Move disk 1 from rod A to rod B
Move disk 4 from rod A to rod C
Move disk 1 from rod B to rod C
Move disk 2 from rod B to rod A
Move disk 1 from rod C to rod A
Move disk 3 from rod B to rod C
Move disk 1 from rod A to rod B
Move disk 2 from rod A to rod C
Move disk 1 from rod B to rod C
```

```
Output:

Tower of Hanoi Solution for 4 disks:

A: [4, 3, 2, 1] B: [] C: []
```

```
Move disk from rod A to rod B
A: [4, 3, 2] B: [1] C: []

Move disk from rod A to rod C
A: [4, 3] B: [1] C: [2]

Move disk from rod B to rod C
A: [4, 3] B: [] C: [2, 1]

Move disk from rod A to rod B
A: [4] B: [3] C: [2, 1]

Move disk from rod C to rod A
A: [4, 1] B: [3] C: [2]

Move disk from rod C to rod B
A: [4, 1] B: [3, 2] C: []

Move disk from rod A to rod B
A: [4] B: [3, 2, 1] C: []

Move disk from rod A to rod C
A: [] B: [3, 2, 1] C: [4]

Move disk from rod B to rod C
A: [] B: [3, 2] C: [4, 1]

Move disk from rod B to rod A
A: [2] B: [3] C: [4, 1]

Move disk from rod C to rod A
A: [2, 1] B: [3] C: [4]

Move disk from rod B to rod C
A: [2, 1] B: [] C: [4, 3]

Move disk from rod A to rod B
A: [2] B: [1] C: [4, 3]
```

```
Move disk from rod A to rod C
A: [] B: [1] C: [4, 3, 2]

Move disk from rod B to rod C
A: [] B: [] C: [4, 3, 2, 1]
```