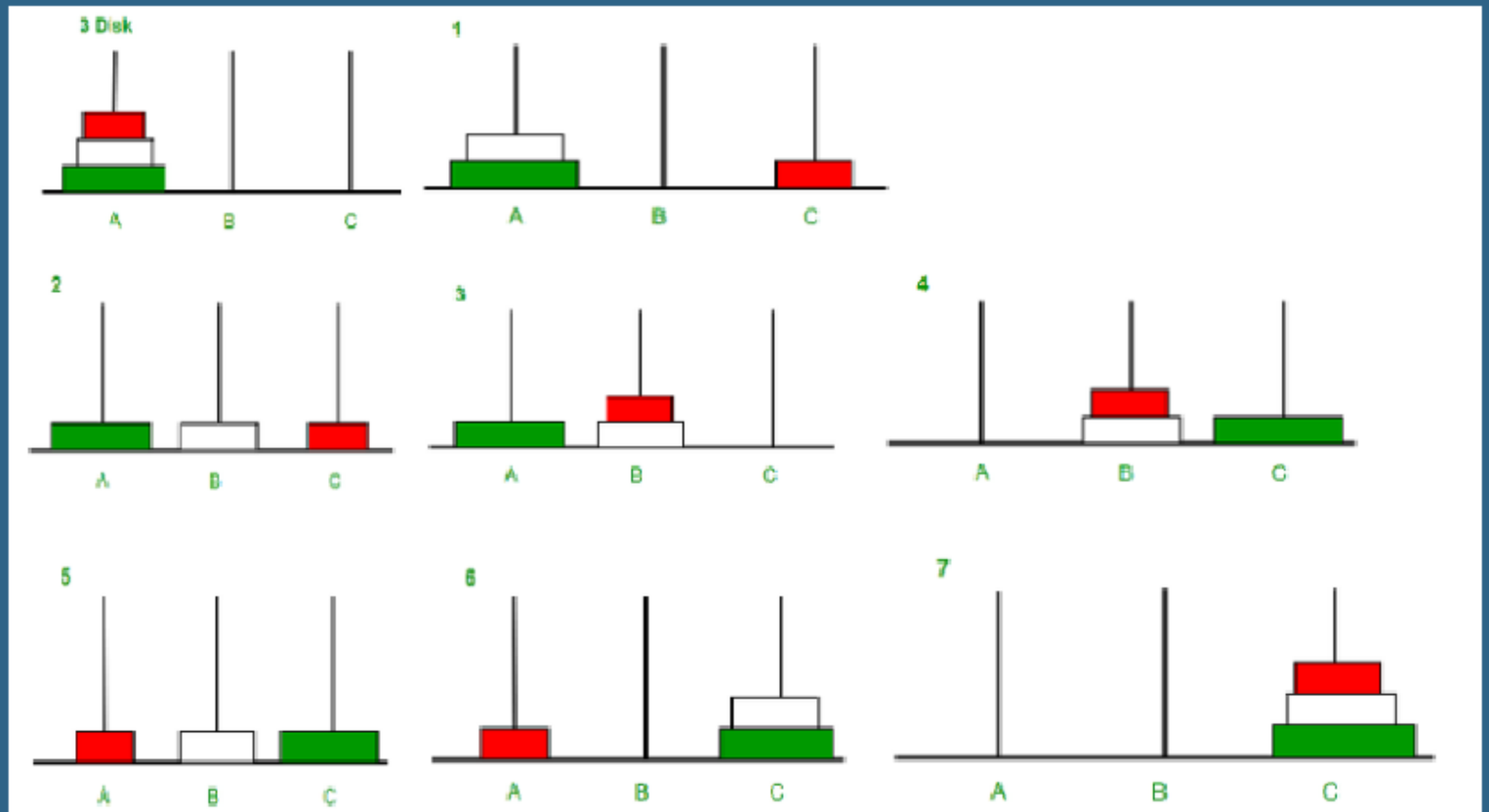
Tower of Hanoi is a mathematical puzzle where we have three rods and n disks. The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules:

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.



# Tower of Hanoi

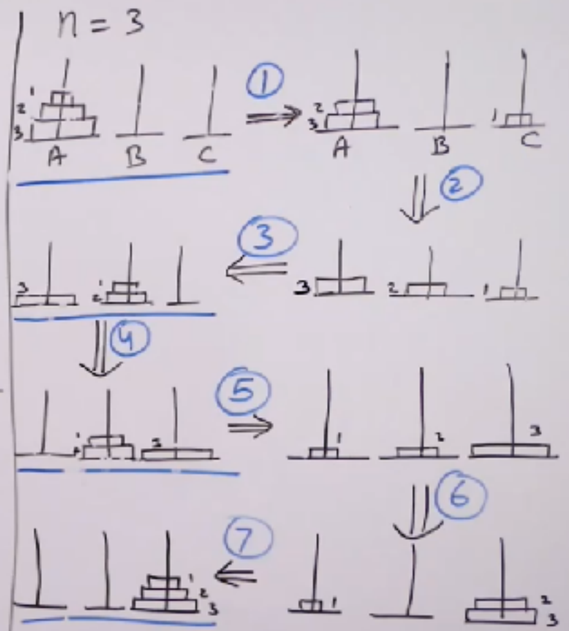
TOH(n, A, B, C)
- → TOH(n-1, A, C, B)
- → Move Disc n from A to C
- → TOH(n-1, B, A, C)

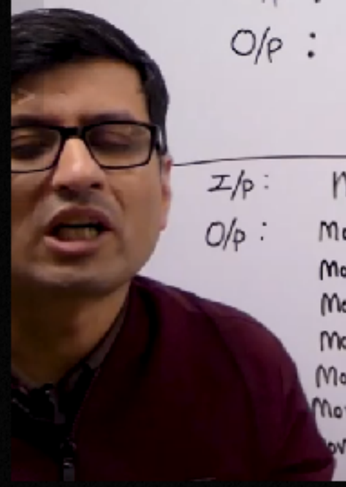
I/p : n = 2

O/p : Move Disc 1 from A to B
Move Disc 2 from A to C
Move Disc 1 from B to C

I/p : n = 3

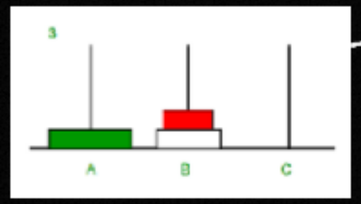
O/p : Move Disc 1 from A to C
Move Disc 2 from A to B
Move Disc 1 from C to B
Move Disc 3 from A to C
Move Disc 1 from B to A
Move Disc 2 from B to C
Move Disc 1 from A to C



A — original
B — Aux
C — Final

(1) From original move (n-1) discs to (B) using C as aux.




(2) Now move Nth disc from A to C:

(3) Nth Disc at its right position.

(4) Transfer B → C using A as Aux.

Thereby recursively calling for the function again.

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

```cpp
// avoid space at the starting of the string in "move disk....."
long long toh(int N, int A, int B, int C) {
long long moves = 0ll;
if (N >= 1) {
    // recursive call to move top disk from "from" to aux in current call
    moves += toh(N - 1, A, C, B);
    printf( "move disk %d from rod %d to rod %d\n" , N, A, B);
    // increment moves
    moves++;

    // recursive call to move top disk from aux to "to" in current call
    moves += toh(N - 1, C, B, A);
}
return moves;

;
```

1) Move n-1 to move (N-1) disc from A → C, B as aux.

TOH(3, 'A', 'B', 'C')

GeeksforGeeks
A computer science portal for geeks

→ TOH(2, 'A', 'C', 'B')

→ TOH(1, 'A', 'B', 'C')

↳ Move 1 from 'A' to 'C'

→ Move 2 from 'A' to 'B'

→ TOH(1, 'C', 'A', 'B')

↳ Move 1 from 'C' to 'B'

→ Move 3 from 'A' to 'C'

→ TOH(2, 'B', 'A', 'C')

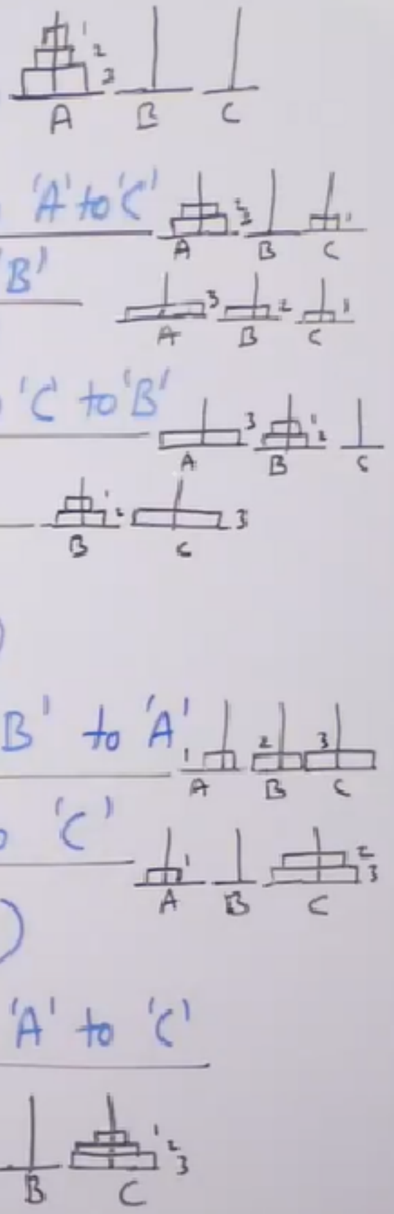
→ TOH(1, 'B', 'C', 'A')

↳ Move 1 from 'B' to 'A'

→ Move 2 from 'B' to 'C'

→ TOH(1, 'A', 'B', 'C')

↳ Move 1 from 'A' to 'C'




① Base case Return when [N == 1]

② Given A, B, C

(N-1) plates as aux

(2.1)

③ Move the Nth plate from A to C.

④ Call for transferring (N-1) plates from B to C using A as aux array