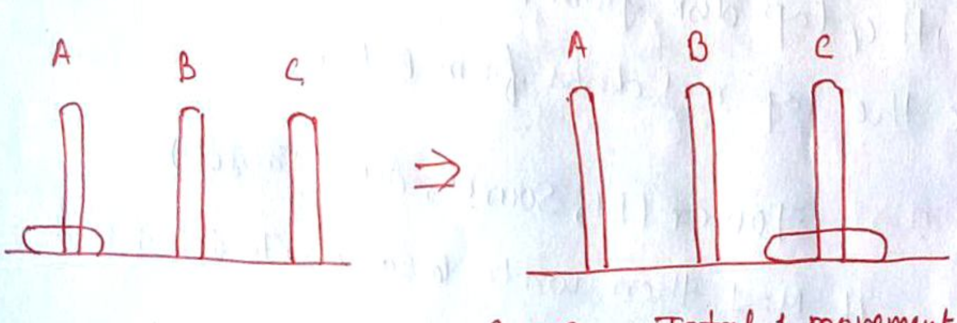
Tower of Hanoi Problem – In this problem, there are n disk of different sizes and there are three rods A, B and C. All the n disks are placed on rod A in such a way that a larger disk is always below a smaller disk. The other two rods are initially empty. The aim is to move the n disks to the rod C using rod B as a temporary storage.

The rules for the movement of disks are as follows:

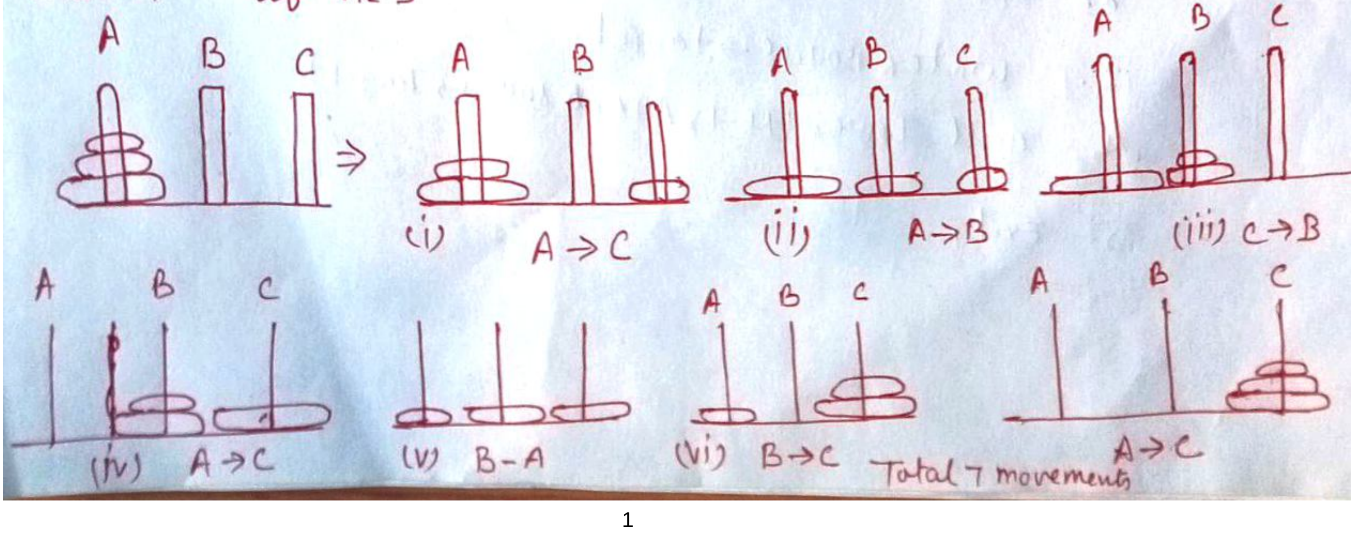
- Only one disk move at a time.
- A larger disk must never be stucked above a smaller one.
- Only the top disk on any rod may be moved to any other rod.

Case 1:- if n = 1



A→C Total 1 movement

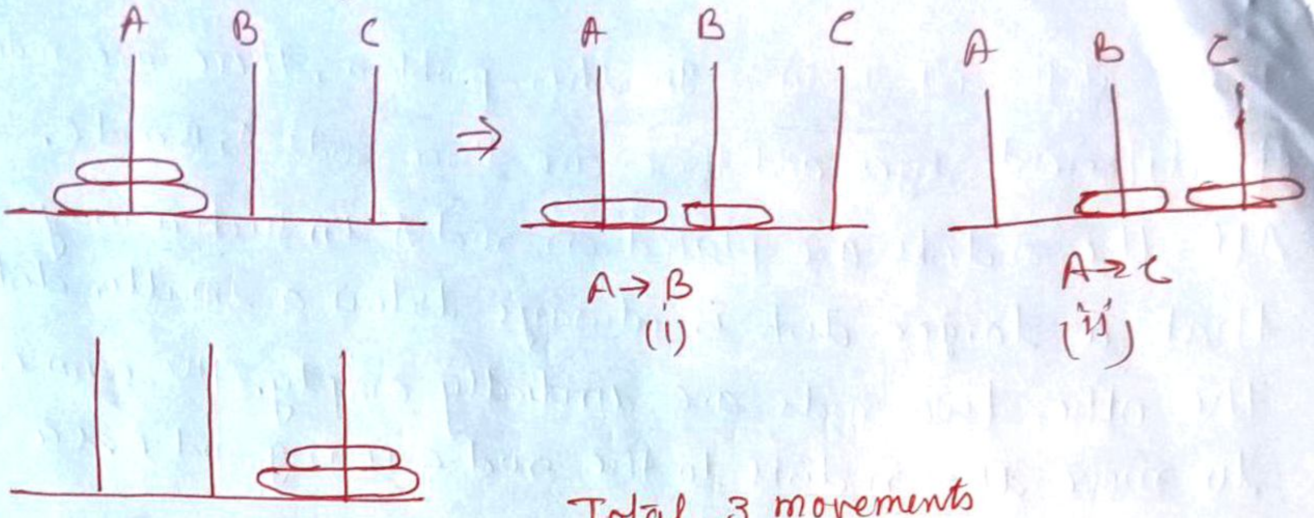
Case 2:- If n = 3



(i) A→C (ii) A→B (iii) C→B

(iv) A→C (v) B-A (vi) B→C A→C Total 7 movements

iii) If (n=2)



Total 3 movements

so if there are n disk then total movement needed $= 2^n - 1$

We use recursion to solve tower of hanoi problem. The solution to tower of Hanoi for $n \geq 1$ disks may be reduced to the following sub problems.

(i) move the top n-1 disks from A to B

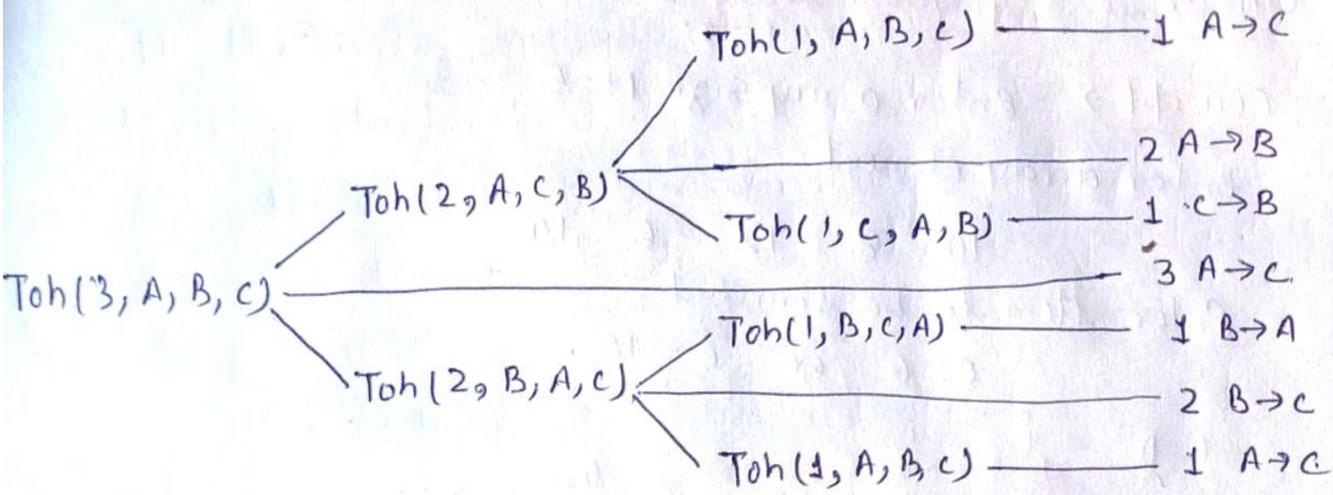
ii) move the top disk from A to C

iii) move the top n-1 disks from B to C

Algorithm → Tower (M, Source, Aux, Target)

```
1. If M=1 then write Source ⇒ Target & Exit
2. call Tower(M-1, source, Target, Aux)
3. write source → target
4. call Tower (M-1, Aux, source, Target)
5. Exit.
```

Example→ Toh(3, A, B, C)



C program→

```
void toh(int N, char S, char A, char T)
{
    if(n>0)
    {
    toh(N-1, S, T, A)
printf("move %d disk from %c to %c", N, S, T);
    tah(N-1, A, S, T);
    }
}

int main()
{
   char S='A', aux='B', T='C';
   int n;
   printf("Enter no. of disk");
   scanf("%d", &N);
   toh(N, S, A, T);
}
```