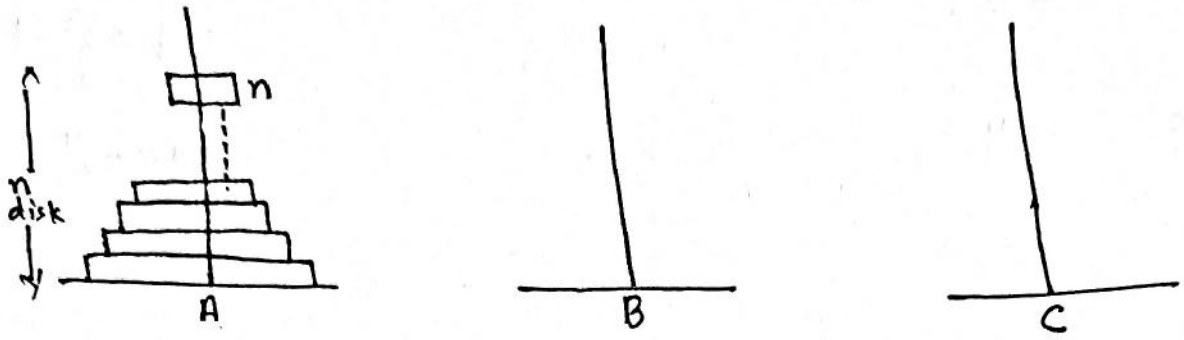
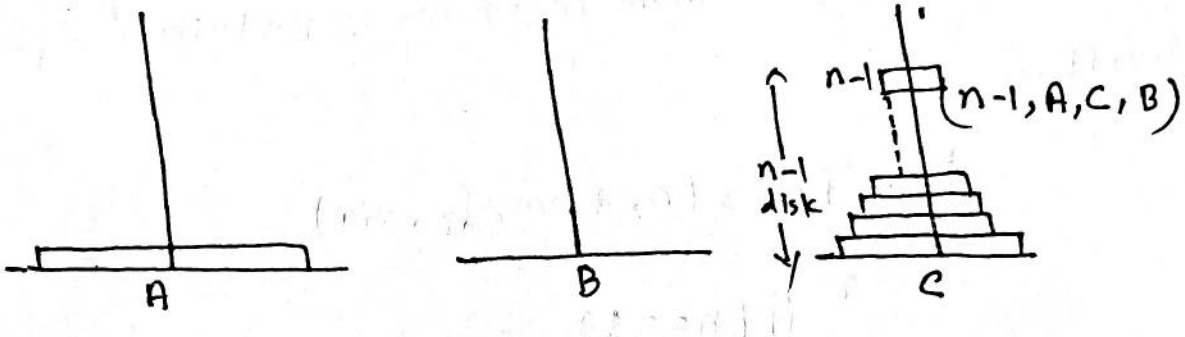
n-disk Tower of Hanoi Problem

n-disk Problem is defined as: (n, A, B, C)

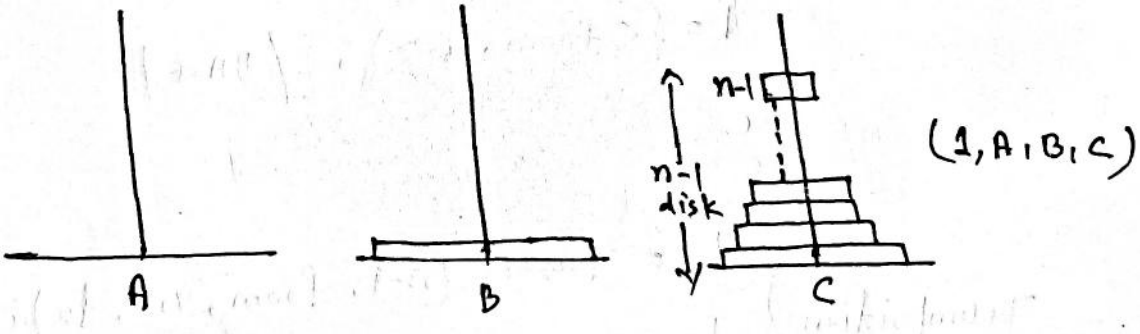


n-disk Problem is also divided into three parts

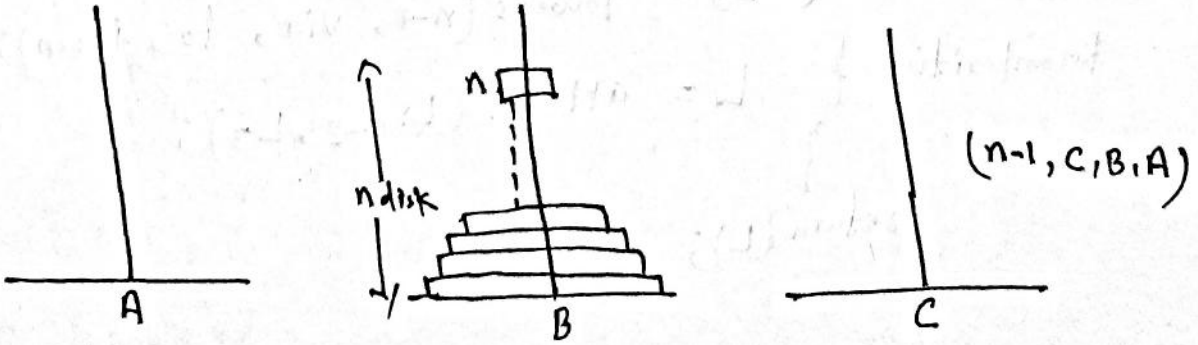
Step 1:



Step 2:



Step 3:



if no. of disk = 1, Minimum number of moves required = $1 = 2^1 - 1$

" = 2, " $= 3 = 2^2 - 1$

" = 3, " $= 7 = 2^3 - 1$

" = 4, " $= 15 = 2^4 - 1$

---

if no. of disk = n, Minimum no. of moves required = $\boxed{2^n - 1}$

Algorithm:

```
List Towers (n, from, to, via)
{
  if (n == 1)
    L = {<from, to>}; /BASE/
  else
  {
Decomposition {
    L1 = Towers (n-1, from, via, to);
    L2 = Towers (1, from, to, via);
    L3 = Towers (n-1, via, to, from);
Recomposition {
    L = append (L1, L2, L3);
  }
  return (L);
}
```

```
# include<stdio.h>
# include<conio.h>

towers (m, from, to, via)
    int m;
    Char from, to, via;
  {
     if (m==1)
       {
         Printf (" Move from %c to %c \n", from, to);
             return;
       }
     else
       {
          towers (m-1, from, via, to);
          Printf (" Move from %c to %c \n", from, to);
          towers (m-1, via, to, from);
       }
  }
void main ()
{ int n;
  Printf ("Give n: ");
  Scanf ("%d", &n);
  towers (n, 'A', 'B', 'C');
  getch();
}.
```