Tower of Hanoi Using Recursion - Quick Revision Notes (Hinglish)

Code ka Logic:

- Function `hanoi(int n, char a, char b, char c)` recursion use karke Tower of Hanoi problem solve karta hai.
- **Base Condition**:
  - Agar `n == 0`, function return karega `0` aur recursion ruk jayega.
- **Recursive Calls**:
  - Pehle `n-1` disks ko source `a` se auxiliary `c` par move karo, destination `b` ka use karke.
  - Phir `nth` disk ko `a` se `b` par move karo.
  - Fir `n-1` disks ko `c` se `b` par move karo, auxiliary `a` ka use karke.
- Is process se saare disks `a` se `b` tak move ho jate hain bina rules tode.

Problem Example:

Problem Statement:
`n` disks aur 3 rods (A, B, C) diye gaye hain. Saare disks ko `A` se `B` par move karna hai `C` ka use karke,
aur ye rules follow karne hain:
1. Ek baar me sirf ek disk move kar sakte hain.
2. Badi disk chhoti disk ke upar nahi rakh sakte.

Example (Agar n = 2 ho):
Steps to move 2 disks from A to B:
1. Disk 1 ko A se C par le jao
2. Disk 2 ko A se B par le jao
3. Disk 1 ko C se B par le jao

```
#include<iostream>
using namespace std;

// Recursion se Tower of Hanoi solve karne ka function
int hanoi(int n, char a, char b, char c) {
    if(n == 0) return 0; // Base case recursion stop karne ke liye
    hanoi(n-1, a, c, b); // n-1 disks ko A se C le jao B ka use karke
    cout << a << " -> " << b << "\n"; // nth disk ko A se B le jao
    hanoi(n-1, b, c, a); // n-1 disks ko C se B le jao A ka use karke
}

int main() {
    int n = 2;
    hanoi(n, 'A', 'B', 'C'); // 2 disks ke liye function call
}
```

Code Ka Dry Run (hanoi(2, 'A', 'B', 'C'))

Function Call	Disk Move
hanoi(2, A, B, C)	Move 1 from A to C
hanoi(1, A, C, B)	Move 2 from A to B
hanoi(1, C, B, A)	Move 1 from C to B

Final Output for hanoi(2, 'A', 'B', 'C'):

A -> C

A -> B

C -> B

Time Complexity:

- Har move ye recurrence relation follow karta hai: T(n) = 2T(n-1) + 1
- Iska time complexity **O(2^n)** hota hai.

Key Takeaways:

- Tower of Hanoi ek classic recursion problem hai.
- Function `n-1` disks ko auxiliary peg pe move karta hai pehle.
- Minimum moves required `n` disks ke liye **(2^n - 1)** hote hain.