Tower of Hanoi - C Program Documentation

The Tower of Hanoi is a classic mathematical puzzle that involves three pegs and a number of disks of different sizes. The puzzle starts with the disks neatly stacked in ascending order of size on one peg, the smallest at the top, making a conical shape. The objective of the puzzle is to move the entire stack to another peg, obeying the following simple rules: 1. Only one disk can be moved at a time. 2. Each move consists of taking the top disk from one stack and placing it on top of another stack. 3. No larger disk may be placed on top of a smaller disk.

Problem Statement:
Write a program in C to solve the Tower of Hanoi puzzle for 'n' disks using recursion. Also calculate and display the total number of moves required.

Algorithm:

```
1. Start.
2. Input the number of disks (n).
3. If n == 1, move the single disk from Source to Destination.
4. Otherwise:
a. Move (n-1) disks from Source to Auxiliary peg.
b. Move the nth disk from Source to Destination peg.
c. Move (n-1) disks from Auxiliary to Destination peg.
5. Repeat recursively until all disks are moved.
6. Total moves required = 2^n - 1.
7. Stop.
```

C Program Code:

```c
#include <stdio.h>
#include <math.h>

// Recursive function to solve Tower of Hanoi
void towerOfHanoi(int n, char source, char auxiliary, char destination) {
    if (n == 1) {
        printf("Move disk 1 from %c to %c\n", source, destination);
        return;
    }
    // Move n-1 disks from source to auxiliary
    towerOfHanoi(n - 1, source, destination, auxiliary);

    // Move the nth disk from source to destination
    printf("Move disk %d from %c to %c\n", n, source, destination);

    // Move n-1 disks from auxiliary to destination
    towerOfHanoi(n - 1, auxiliary, source, destination);
}

int main() {
    int n;
    printf("Enter number of disks: ");
    scanf("%d", &n);

    long long totalMoves = pow(2, n) - 1;  // Formula for total moves
    printf("\nTotal moves required: %lld\n", totalMoves);

    printf("\nTower of Hanoi solution:\n");
    towerOfHanoi(n, 'A', 'B', 'C');

    return 0;
}
```

Sample Output 1 (n = 2):

```
Enter number of disks: 2
Total moves required: 3
Move disk 1 from A to B
Move disk 2 from A to C
Move disk 1 from B to C
```

Sample Output 2 (n = 3):

```
Enter number of disks: 3
Total moves required: 7
Move disk 1 from A to C
Move disk 2 from A to B
Move disk 1 from C to B
Move disk 3 from A to C
Move disk 1 from B to A
Move disk 2 from B to C
Move disk 1 from A to C
```

Sample Output 3 (n = 4):

```
Enter number of disks: 4
Total moves required: 15
Move disk 1 from A to B
Move disk 2 from A to C
Move disk 1 from B to C
Move disk 3 from A to B
Move disk 1 from C to A
Move disk 2 from C to B
Move disk 1 from A to B
Move disk 4 from A to C
Move disk 1 from B to C
Move disk 2 from B to A
Move disk 1 from C to A
Move disk 3 from B to C
Move disk 1 from A to B
Move disk 2 from A to C
Move disk 1 from B to C
```

Conclusion:

The Tower of Hanoi program demonstrates the power of recursion in solving complex problems by breaking them down into smaller sub-problems. The minimum number of moves required to solve the puzzle is always (2^n - 1), where 'n' is the number of disks. This project also illustrates how mathematical formulas and recursive algorithms can be implemented effectively in C programming.