**Tower of Hanoi**

**Objective**

To implement the Tower of Hanoi puzzle using a recursive approach in C.

**Theory**

The Tower of Hanoi is a classic puzzle consisting of three rods and a number of disks of different sizes, which can slide onto any rod. The puzzle starts with all the disks stacked in ascending order of size on one rod, the smallest at the top. The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules:

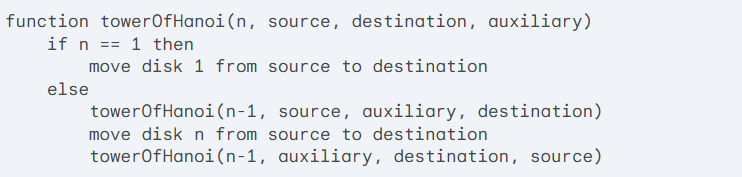
1. Only one disk can be moved at a time.
2. Each move consists of taking the uppermost disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.
3. No disk may be placed on top of a smaller disk.

**Recursive Solution:**

The Tower of Hanoi puzzle can be solved recursively using the following steps:

1. **Base Case:** If there is only one disk, move it from the source peg to the destination peg.
2. **Recursive Case:**
   * Move the top n-1 disks from the source peg to the auxiliary peg.
   * Move the nth disk from the source peg to the destination peg.
   * Move the n-1 disks from the auxiliary peg to the destination peg.

**Algorithm:**

****

**C Implementation:**

#include <stdio.h>

// Function to perform Tower of Hanoi

void towerOfHanoi(int n, char source, char destination, char auxiliary) {

    // Base case: if only one disk, move it directly

    if (n == 1) {

        printf("Move disk 1 from %c to %c\n", source, destination);

        return;

    }

    // Step 1: Move top n-1 disks from source to auxiliary

    towerOfHanoi(n - 1, source, auxiliary, destination);

    // Step 2: Move the nth disk from source to destination

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

    // Step 3: Move the n-1 disks from auxiliary to destination

    towerOfHanoi(n - 1, auxiliary, destination, source);

}

int main() {

    int n;  // Number of disks

    printf("Enter the number of disks: ");

    scanf("%d", &n);

    // Call the Tower of Hanoi function

    towerOfHanoi(n, 'A', 'C', 'B');  // A is the source, C is the destination, and B is the auxiliary

    return 0;

}

}

**Analysis:**

**Time Complexity:** The time complexity of the Tower of Hanoi algorithm is exponential, O(2^n). This is because for each disk, the function makes two recursive calls, doubling the number of operations with each level of recursion.

**Space Complexity:** The space complexity of the Tower of Hanoi algorithm is O(n), due to the recursion stack. The depth of the recursion tree is n, and each recursive call adds a constant amount of space to the stack.