The Tower of Hanoi algorithm is a classic recursive algorithm used to solve the Tower of Hanoi puzzle, which involves moving a stack of disks from one peg to another, with the constraint that a larger disk cannot be placed on top of a smaller disk. The algorithm has a time complexity of O(2^n), where n is the number of disks.

Here's how the Tower of Hanoi algorithm works:

1. If there is only one disk, move it from the source peg to the destination peg.
2. If there are more than one disk, move n-1 disks from the source peg to the auxiliary peg using the destination peg as a temporary peg.
3. Move the largest disk from the source peg to the destination peg.
4. Move the n-1 disks from the auxiliary peg to the destination peg using the source peg as a temporary peg.

Each recursive call to the algorithm reduces the problem size by one, so the algorithm is called 2^n times, where n is the number of disks. Therefore, the time complexity of the Tower of Hanoi algorithm is O(2^n).

Here's an example of how the algorithm would work for a stack of 3 disks:

css

Move disk 1 from source peg to destination peg

Move disk 2 from source peg to auxiliary peg

Move disk 1 from destination peg to auxiliary peg

Move disk 3 from source peg to destination peg

Move disk 1 from auxiliary peg to source peg

Move disk 2 from auxiliary peg to destination peg

Move disk 1 from source peg to destination peg

In this example, the algorithm required 7 moves to solve the puzzle, which matches the 2^n - 1 formula for a stack of 3 disks, where 2^3 - 1 = 7.