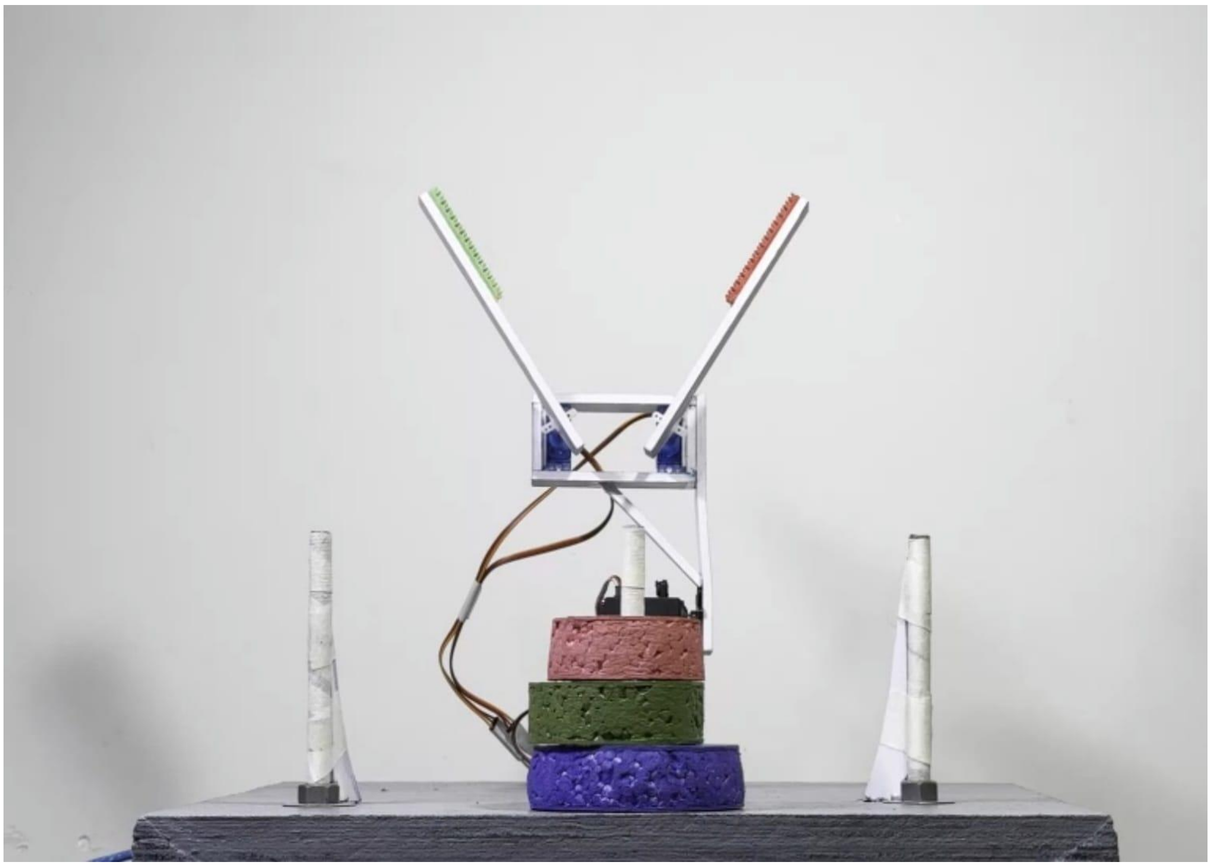
Shri Ramdeobaba College of Engineering and Management, Nagpur

Embedded Club

HanoiHelper: The Puzzle-Solving Prodigy

Project Head: Mohit Agrawal, 3rd Yr EN

Project Member 1: Aneesh Thakre, 2nd Yr ECS
Project Member 2: Sarvesh Shirulkar, 2nd Yr ECS



The **Tower of Hanoi** is a mathematical game or puzzle consisting of three rods and a number of disks of various diameters, which can slide onto any rod. The puzzle begins with the disks stacked on one rod in order of decreasing size, the smallest at the top, thus approximating a conical shape. The objective of the puzzle is to move the entire stack to one of the other rods, obeying the following rules:

1. Only one disk may be moved at a time.
2. Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack or on an empty rod.
3. No disk may be placed on top of a disk that is smaller than it.

With three disks, the puzzle can be solved in seven moves. The minimal number of moves required to solve a Tower of Hanoi puzzle is $2^n - 1$, where n is the number of disks.

Recursive Algorithm :

The key to solving a problem recursively is to recognize that it can be broken down into a collection of smaller sub-problems, to each of which *that same general solving procedure that we are seeking* applies, and the total solution is then found in some *simple* way from those sub-problems' solutions. Each of these created sub-problems being "smaller" guarantees that the base case(s) will eventually be reached. Thence, for the Towers of Hanoi:

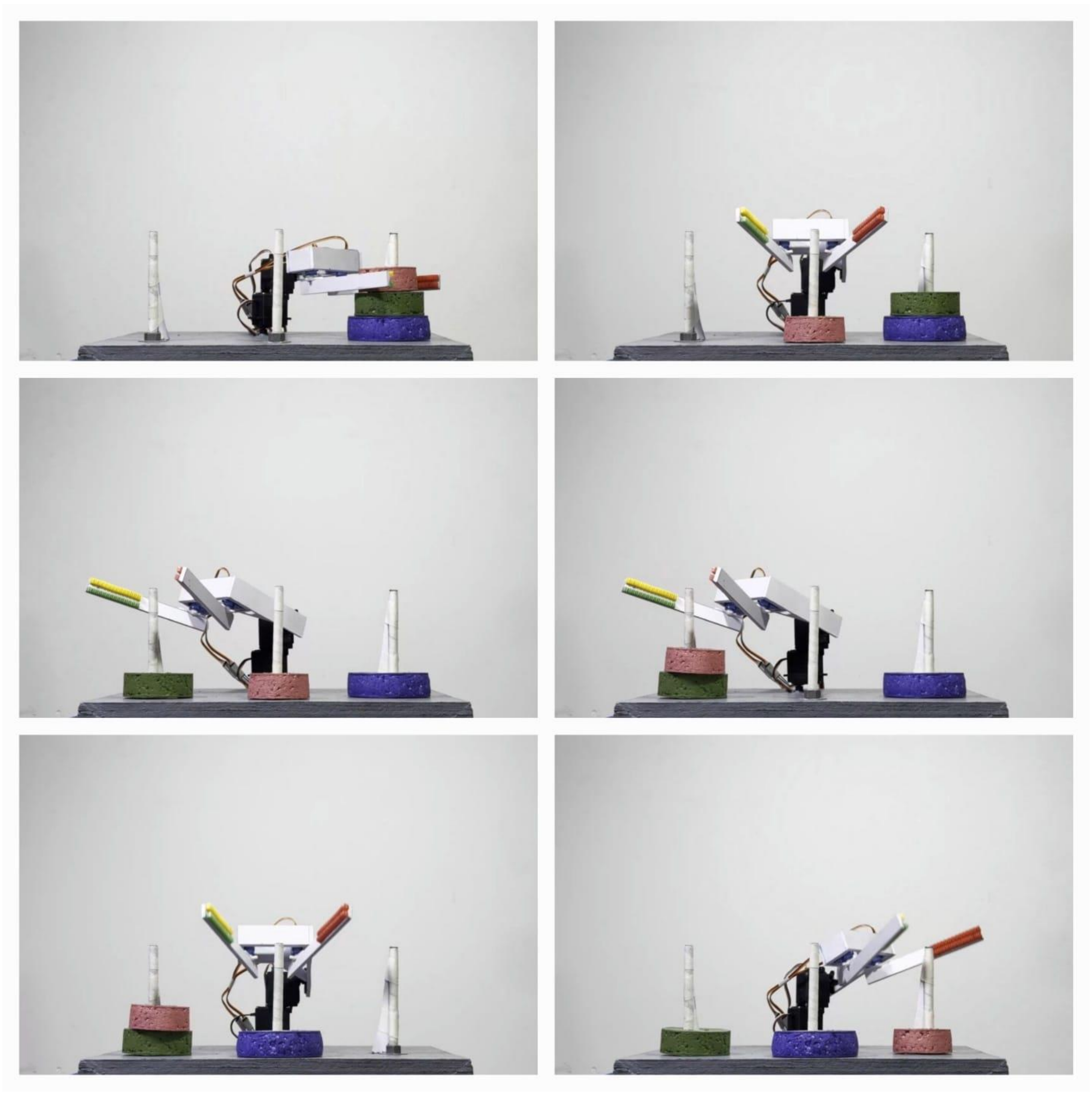
1. label the pegs A, B, C,
2. let n be the total number of disks,
3. number the disks from 1 (smallest, topmost) to n (largest, bottom-most).

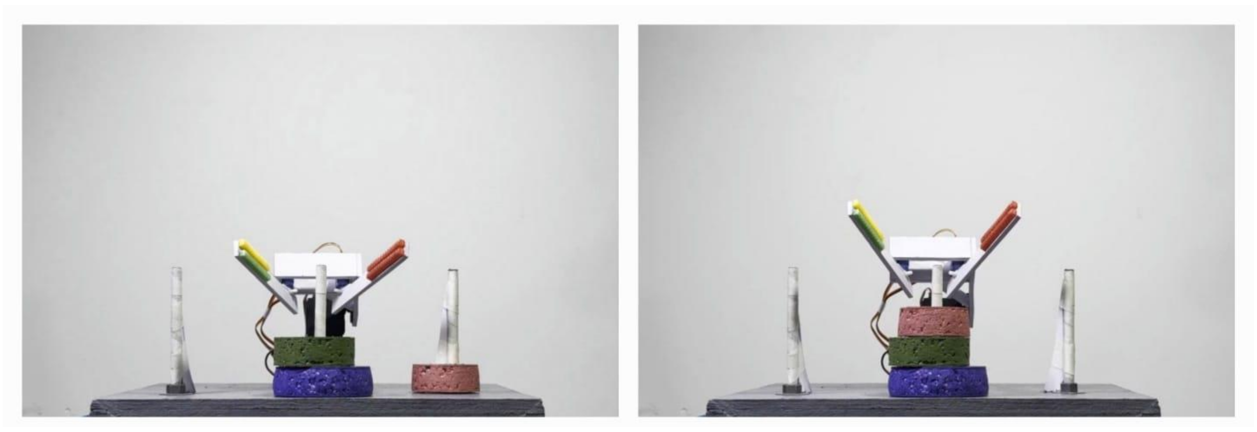
Assuming all n disks are distributed in valid arrangements among the pegs; assuming there are m top disks on a *source* peg, and all the rest of the disks are larger than m, so they can be safely ignored; to move m disks from a source peg to a *target* peg using a *spare* peg, without violating the rules:

1. Move $m - 1$ disks from the **source** to the **spare** peg, by *the same general solving procedure*. Rules are not violated, by assumption. This leaves the disk m as a top disk on the source peg.
2. Move the disk m from the **source** to the **target** peg, which is guaranteed to be a valid move, by the assumptions — *a simple step*.
3. Move the $m - 1$ disks that we have just placed on the spare, from the **spare** to the **target** peg by *the same general solving procedure*, so they are placed on top of the disk m without violating the rules.
4. The base case is to move *0* disks (in steps 1 and 3), that is, do nothing—which obviously does not violate the rules.

The full Tower of Hanoi solution then consists of moving n disks from the source peg A to the target peg C, using B as the spare peg.

Solving Algorithm by Arm :



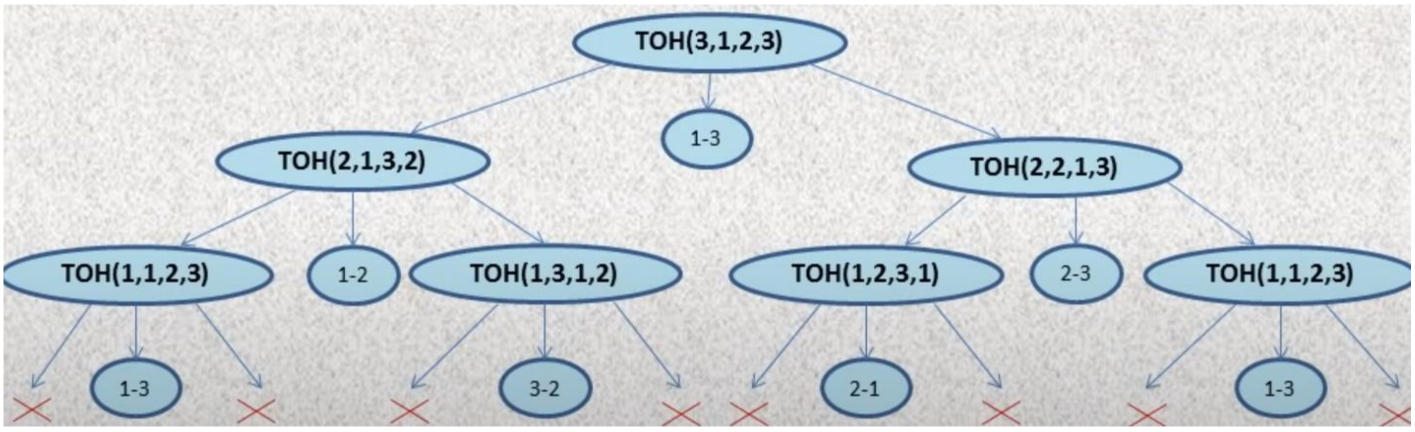


Recursive Code :

```
static void hanoiAlgo(int n, char from, char to, char using){
    {
      if(n==0){
        System.out.println("No Disks");
      }
      else if(n == 1){
        System.out.println("Move disk "+ n +" from "+ from + " to "+ to );
      }
      else{
        hanoiAlgo(n-1, from, using, to);
        System.out.println("Move disk "+ n + " from " + from + " to " + to);
        hanoiAlgo(n-1, using, to, from);
      }
    }
  }
```

Output :

```
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
```



Real-life Implications :

1. Educational Tool: Teaches recursion, loops, and signal propagation in DSA, microprocessor programming, and EDC, respectively.

2. Industrial Automation: Simulates material handling, assembly, and packaging processes, showcasing automation precision.

3. Research and Development: Used as a benchmark for algorithm testing and optimization studies.

4. Assistive Technology: Transformed into an interactive game for cognitive rehabilitation and skill improvement.