Three views for the recursion solution of Hanoi problem

1. **Problem View:**
   Break down the problem size into small pieces. Figure out the way to reassemble the solution for the small pieces.

   Function Signature: `void Solve(int numofDisks, String start, String tmp, String end)`

   Goal: `Solve(n, a, b, c)`

   ![Diagram of Hanoi problem solution]

   1. Solve(n-1, a, c, b)
   2. Move disk#n from a to c
   3. Solve(n-1, b, a, c)
2. **Code view:**
   Translate the idea into recursion program.

Code:
------------------------------------------------------------------------------------------
public static int counter;

/* choose 1 disk as the base case */
public static void solveTowers(int numberOfDisks, String startPole, String tempPole, String endPole) {
    /* base case, recursion terminated condition */
    if (numberOfDisks == 1) {
        System.out.println("Move Disk "+numberOfDisks+" from "+ startPole +" to "+ endPole);
        counter++;
        return;
    }
    /* step 1 */
    solveTowers(numberOfDisks-1, startPole, endPole, tempPole);
    /* step 2 */
    System.out.println("Move Disk "+numberOfDisks+" from "+ startPole +" to "+ endPole);
    counter++;
    /* step 3 */
    solveTowers(numberOfDisks-1, tempPole, startPole, endPole);
}
------------------------------------------------------------------------------------------

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
total moves = 7
------------------------------------------------------------------------------
3. Call Stack View
How the program is being executed?

```
Solve(1, a, b, c)
```

```
Solve(2, a, c, b)
Solve(3, a, b, c)
```

```
Solve(1, a, b, c)
```

```
Solve(2, a, c, b)
Solve(3, a, b, c)
```

```
Solve(1, c, a)
```

```
Solve(2, a, c, b)
Solve(3, a, b, c)
```

```
Solve(1, b, c, a)
```

```
Solve(2, b, a, c)
Solve(3, a, b, c)
```

```
Solve(1, a, b, c)
Solve(2, b, a, c)
Solve(3, a, b, c)
```

```
Solve(1, a, b, c)
```

```
Solve(2, a, c, b)
Solve(3, a, b, c)
```

```
Solve(1, c, a)
```

```
Solve(2, a, c, b)
Solve(3, a, b, c)
```

```
Solve(1, b, c, a)
```

```
Solve(2, b, a, c)
Solve(3, a, b, c)
```

```
Solve(1, a, b, c)
```

```
Solve(2, a, c, b)
Solve(3, a, b, c)
```

```
Solve(1, c, a)
```

```
Solve(2, a, c, b)
Solve(3, a, b, c)
```

```
Solve(1, b, c, a)
```

```
Solve(2, b, a, c)
Solve(3, a, b, c)
```