TOWER OF HANOI (General Method)

```
package recursion;

public class TOH {

        public static void main(String[] args) {

    int numberOfDisks = 3;

    char source = 'A';

    char auxiliary = 'B';

    char destination = 'C';


    System.out.println("Steps to solve Tower of Hanoi with " +
numberOfDisks + " disks:");

    solveTowerOfHanoi(numberOfDisks, source, auxiliary,
destination);

    }


  public static void solveTowerOfHanoi(int n, char source, char
auxiliary, char destination) {

    if (n == 1) {

      System.out.println("Move disk from " + source + " to " +
destination);

      return;

    } else {
```

```
                solveTowerOfHanoi(n - 1, source, destination,
auxiliary);

                solveTowerOfHanoi(1, source, auxiliary, destination );

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

            }

        }

    }
```

TOWER OF HANOI (Using Stack)

```
package recursion;
import java.util.Stack;
public class TowerOfHanoi {
        public static void towerOfHanoi(int numDisks, Stack<Integer>
source, Stack<Integer> auxiliary, Stack<Integer> destination) {
            if (numDisks == 1) {
                destination.push(source.pop());
                System.out.println("Move disk 1 from source to destination");
                return;
            }

            towerOfHanoi(numDisks - 1, source, destination, auxiliary);
            destination.push(source.pop());
            System.out.println("Move disk " + numDisks + " from source to
destination");
            towerOfHanoi(numDisks - 1, auxiliary, source, destination);
        }

        public static void main(String[] args) {
            int numDisks = 3;
            Stack<Integer> source = new Stack<>();
            Stack<Integer> auxiliary = new Stack<>();
```

```
        Stack<Integer> destination = new Stack<>();

        // Initialize source stack with disks
        for (int i = numDisks; i >= 1; i--) {
            source.push(i);
        }

        System.out.println("Initial configuration:");
        System.out.println("Source: " + source);
        System.out.println("Auxiliary: " + auxiliary);
        System.out.println("Destination: " + destination);

        // Solve Tower of Hanoi problem
        towerOfHanoi(numDisks, source, auxiliary, destination);

        System.out.println("Final configuration:");
        System.out.println("Source: " + source);
        System.out.println("Auxiliary: " + auxiliary);
        System.out.println("Destination: " + destination);
    }
}
```