Solution 1   Solution 2   Solution 3

```swift
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

class Program {
    // Upper Bound: (O(n * (2n)!)/(n!(n + 1)!)) time | O(n) space
    func numberOfBinaryTreeTopologies(_ n: Int) -> Int {
        if n == 0 {
            return 1
        }

        var numberOfTopologies = 0

        for leftTreeSize in 0 ..< n {
            let rightTreeSize = n - 1 - leftTreeSize

            let leftNumberOfTopologies = numberOfBinaryTreeTopologies(leftTreeSize)
            let rightNumberOfTopologies = numberOfBinaryTreeTopologies(rightTreeSize)
            numberOfTopologies += leftNumberOfTopologies * rightNumberOfTopologies
        }

        return numberOfTopologies
    }
}
```

```swift
class Program {
    // Upper Bound: (O(n * (2n)!)/(n!(n + 1)!)) time | O(n) space
    func numberOfBinaryTreeTopologies(_ n: Int) -> Int {
        if n == 0 {
            return 1
        }

        var numberOfTopologies = 0

        for leftTreeSize in 0 ..< n {
            let rightTreeSize = n - 1 - leftTreeSize

            let leftNumberOfTopologies = numberOfBinaryTreeTopologies(leftTreeSize)
            let rightNumberOfTopologies = numberOfBinaryTreeTopologies(rightTreeSize)
```