```
1  // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3  using System.Collections.Generic;
4
5  public class Program {
6    // O(n^2) time | O(n) space
7    public static int NumberOfBinaryTreeTopologies(int n) {
8      Dictionary<int, int> cache = new Dictionary<int, int>();
9      cache.Add(0, 1);
10     return NumberOfBinaryTreeTopologies(n, cache);
11   }
12
13   public static int NumberOfBinaryTreeTopologies(int n, Dictionary<int, int> cache) {
14     if (cache.ContainsKey(n)) {
15       return cache[n];
16     }
17     int numberOfTrees = 0;
18     for (int leftTreeSize = 0; leftTreeSize < n; leftTreeSize++) {
19       int rightTreeSize = n - 1 - leftTreeSize;
20       int numberOfLeftTrees = NumberOfBinaryTreeTopologies(leftTreeSize, cache);
21       int numberOfRightTrees = NumberOfBinaryTreeTopologies(rightTreeSize, cache);
22       numberOfTrees += numberOfLeftTrees * numberOfRightTrees;
23     }
24     cache.Add(n, numberOfTrees);
25     return numberOfTrees;
26   }
27 }
28
```