

Our Solution(s)

Run Code

Solution 1Solution 2Solution 3

```
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         List<int> cache = new List<int>();
9         cache.Add(1);
10        for (int m = 1; m < n + 1; m++) {
11            int numberOfTrees = 0;
12            for (int leftTreeSize = 0; leftTreeSize < m; leftTreeSize++) {
13                int rightTreeSize = m - 1 - leftTreeSize;
14                int numberOfLeftTrees = cache[leftTreeSize];
15                int numberOfRightTrees = cache[rightTreeSize];
16                numberOfTrees += numberOfLeftTrees * numberOfRightTrees;
17            }
18            cache.Add(numberOfTrees);
19        }
20        return cache[n];
21    }
22 }
23
```

Your Solutions

Run Code

Solution 1Solution 2Solution 3

```
1 public class Program {
2     public static int NumberOfBinaryTreeTopologies(int n) {
3         // Write your code here.
4         return -1;
5     }
6 }
7
```

Our Tests

```
1 public class Program {
2     // ...
3     public static int NumberOfBinaryTreeTopologies(int n) {
4         // ...
5     }
6 }
7
```

Custom Output

Submit Code

```
10  # Write down the first 1000 values of the sequence in a list
11  x = []
12
13  # Loop
14  for i in range(1000):
15      # Write down the first 1000 values of the sequence in a list
16      x.append(x[i-1] + x[i-2])
17
18  # Print the list
19  print(x)
```

Run or submit code when you're ready.