Our Solution(s) Run Code

Solution 1 Solution 2 Solution 3

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

class Program {
    // O(n^2) time | O(1) space
    func numberOfBinaryTreeTopologies(_ n: Int) -> Int {
        var cache = [0: 1]
        return numberOfBinaryTreeTopologiesHelper(n, &cache)
    }

    func numberOfBinaryTreeTopologiesHelper(_ n: Int, _ cache: inout
        if let cachedValue = cache[n] {
            return cachedValue
        }

        var numberOfTopologies = 0

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

            let leftNumberOfTopologies = numberOfBinaryTreeTopologiesH
            let rightNumberOfTopologies = numberOfBinaryTreeTopologies
            numberOfTopologies += leftNumberOfTopologies * rightNumber
        }

        cache[n] = numberOfTopologies
        return numberOfTopologies
    }
}
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```swift
class Program {
    func numberOfBinaryTreeTopologies(_ n: Int) -> Int {
        // Write your code here.
        return -1
    }
}
```

Our Tests

Custom Output Submit Code

Run or submit code when you're ready.