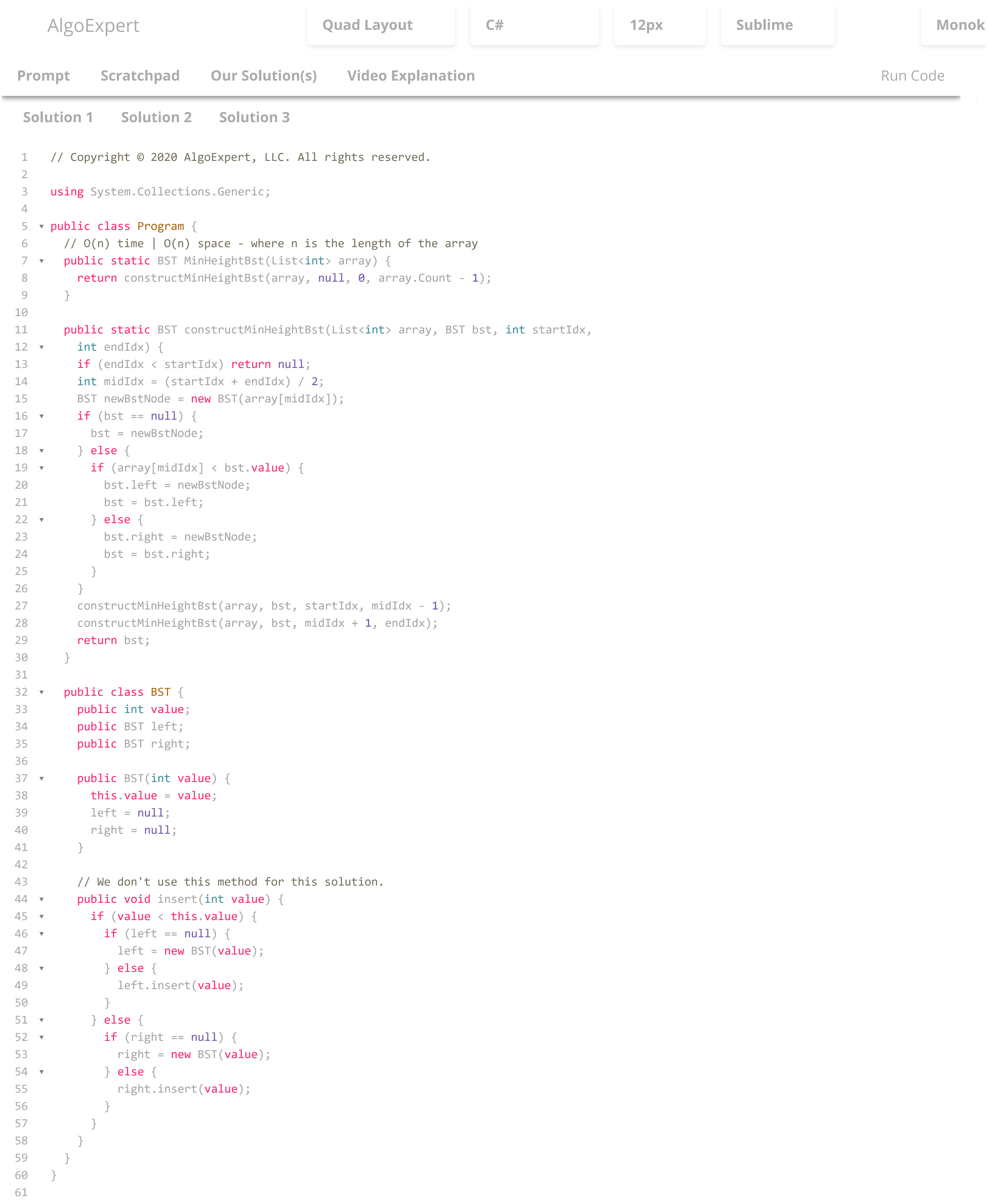

```csharp
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

using System.Collections.Generic;

public class Program {
  // O(n) time | O(n) space - where n is the length of the array
  public static BST MinHeightBst(List<int> array) {
    return constructMinHeightBst(array, null, 0, array.Count - 1);
  }

  public static BST constructMinHeightBst(List<int> array, BST bst, int startIdx,
    int endIdx) {
    if (endIdx < startIdx) return null;
    int midIdx = (startIdx + endIdx) / 2;
    BST newBstNode = new BST(array[midIdx]);
    if (bst == null) {
      bst = newBstNode;
    } else {
      if (array[midIdx] < bst.value) {
        bst.left = newBstNode;
        bst = bst.left;
      } else {
        bst.right = newBstNode;
        bst = bst.right;
      }
    }
    constructMinHeightBst(array, bst, startIdx, midIdx - 1);
    constructMinHeightBst(array, bst, midIdx + 1, endIdx);
    return bst;
  }

  public class BST {
    public int value;
    public BST left;
    public BST right;

    public BST(int value) {
      this.value = value;
      left = null;
      right = null;
    }

    // We don't use this method for this solution.
    public void insert(int value) {
      if (value < this.value) {
        if (left == null) {
          left = new BST(value);
        } else {
          left.insert(value);
        }
      } else {
        if (right == null) {
          right = new BST(value);
        } else {
          right.insert(value);
        }
      }
    }
  }
}
```