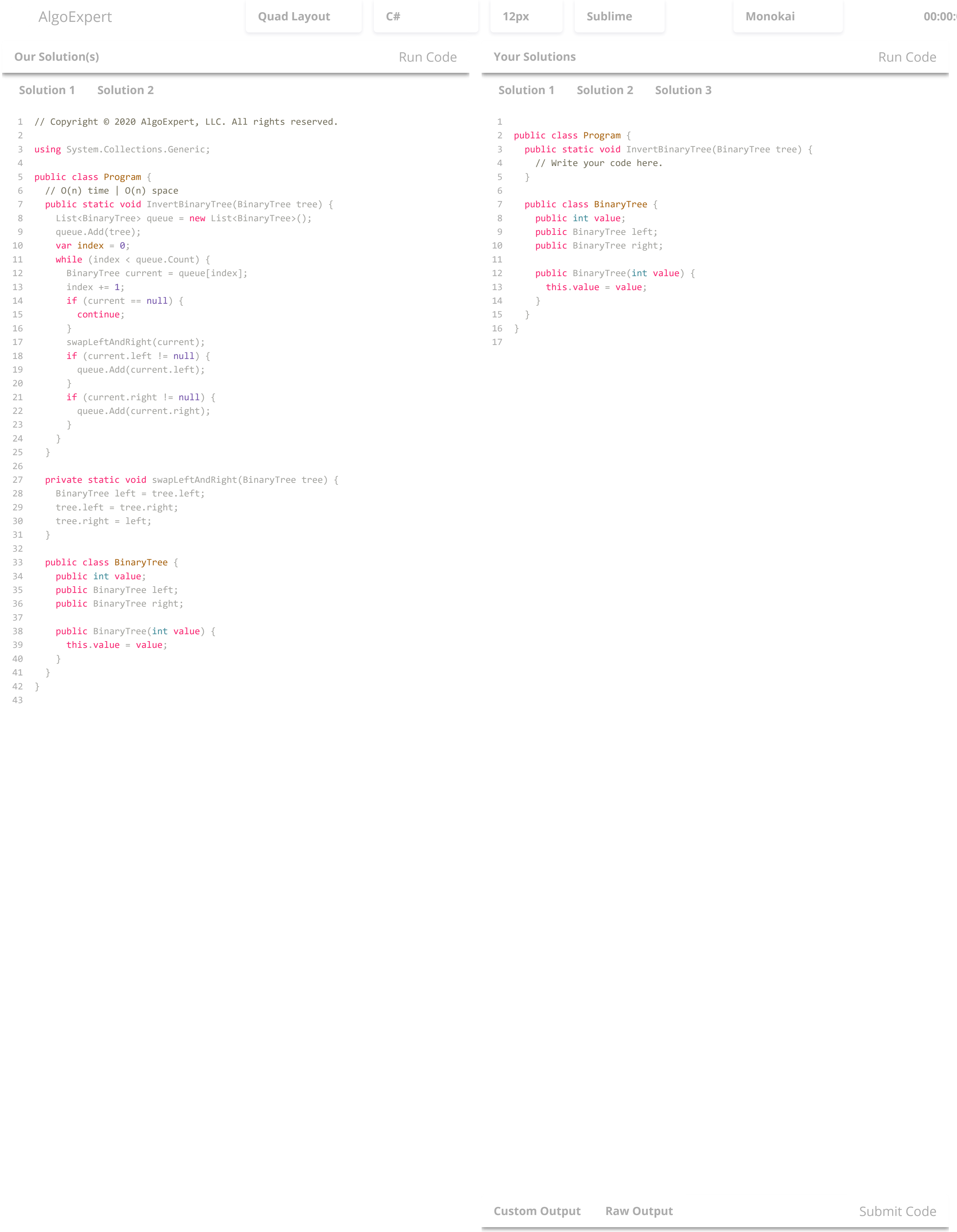

```
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

using System.Collections.Generic;

public class Program {
  // O(n) time | O(n) space
  public static void InvertBinaryTree(BinaryTree tree) {
    List<BinaryTree> queue = new List<BinaryTree>();
    queue.Add(tree);
    var index = 0;
    while (index < queue.Count) {
      BinaryTree current = queue[index];
      index += 1;
      if (current == null) {
        continue;
      }
      swapLeftAndRight(current);
      if (current.left != null) {
        queue.Add(current.left);
      }
      if (current.right != null) {
        queue.Add(current.right);
      }
    }
  }

  private static void swapLeftAndRight(BinaryTree tree) {
    BinaryTree left = tree.left;
    tree.left = tree.right;
    tree.right = left;
  }

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

    public BinaryTree(int value) {
      this.value = value;
    }
  }
}

```

```

public class Program {
  public static void InvertBinaryTree(BinaryTree tree) {
    // Write your code here.
  }

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

    public BinaryTree(int value) {
      this.value = value;
    }
  }
}

```

Run or submit code when you're ready.