Our Solution(s) Run Code

Solution 1 Solution 2

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

using System;
using System.Collections.Generic;

public class Program {
  // Upper Bound: O(n^2*n!) time | O(n*n!) space
  // Roughly: O(n*n!) time | O(n*n!) space
  public static List<List<int> > GetPermutations(List<int> array
    List<List<int> > permutations = new List<List<int> >();
    GetPermutations(array, new List<int>(), permutations);
    return permutations;
  }

  public static void GetPermutations(List<int> array, List<int>
    List<List<int> > permutations) {
    if (array.Count == 0 && currentPermutation.Count > 0) {
      permutations.Add(currentPermutation);
    } else {
      for (int i = 0; i < array.Count; i++) {
        List<int> newArray = new List<int>(array);
        newArray.RemoveAt(i);
        List<int> newPermutation = new List<int>(currentPermutat
        newPermutation.Add(array[i]);
        GetPermutations(newArray, newPermutation, permutations);
      }
    }
  }
}

```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```
using System.Collections.Generic;

public class Program {
  public static List<List<int> > GetPermutations(List<int> array
    // Write your code here.
    return null;
  }
}

```

Our Tests Custom Output Submit Code

```
using System.Linq;
using System.Collections.Generic;

public class Program{Test} {
  [Test]
  public void Test0(){
    int[][] input = new int[][]{};
    int[][]int[][] = parse + Program.GetPermutations(input);
    While.Assert(parse.Count == 0);
  }

  [Test]
  public void Test1(){
    int[][] input = new int[][]{
      {
      },
    };
    int[][]int[][] = parse + Program.GetPermutations(input);
    While.Assert(parse.Count == 1);
    While.Assert(int[][]equals(parse, new int[][]{
      {
      },
    }));
```

Run or submit code when you're ready.