Our Solution(s) Run Code

Solution 1 Solution 2

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

import java.util.*;

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<Integer>> getPermutations(List<Integer
    List<List<Integer>> permutations = new ArrayList<List<Integ
    getPermutations(array, new ArrayList<Integer>(), permutatio
    return permutations;
  }

  public static void getPermutations(
      List<Integer> array, List<Integer> currentPermutation, Li
    if (array.size() == 0 && currentPermutation.size() > 0) {
      permutations.add(currentPermutation);
    } else {
      for (int i = 0; i < array.size(); i++) {
        List<Integer> newArray = new ArrayList<Integer>(array);
        newArray.remove(i);
        List<Integer> newPermutation = new ArrayList<Integer>(cu
        newPermutation.add(array.get(i));
        getPermutations(newArray, newPermutation, permutations);
      }
    }
  }
}

```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```java
import java.util.*;

class Program {
  public static List<List<Integer>> getPermutations(List<Integer
    // Write your code here.
    return null;
  }
}

```

Our Tests Custom Output Submit Code

```
class ProgramTest {
  @Test
  public void TestCase1() {
    List<Integer> input = new ArrayList<Integer>();
    List<List<Integer>> parms = Program.getPermutations(input);
    Assert.fail(parms.size() == 0);
  }

  @Test
  public void TestCase2() {
    List<Integer> input = new ArrayList<Integer>(Arrays.asList(
    List<List<Integer>> parms = Program.getPermutations(input);
    Assert.fail(parms.size() == 1);
    Assert.fail(List<Integer> parms, new ArrayList<Integer>(Arra
  }

  @Test
  public void TestCase3() {
    List<Integer> input = new ArrayList<Integer>(Arrays.asList(
    List<List<Integer>> parms = Program.getPermutations(input);
```

Run or submit code when you're ready.