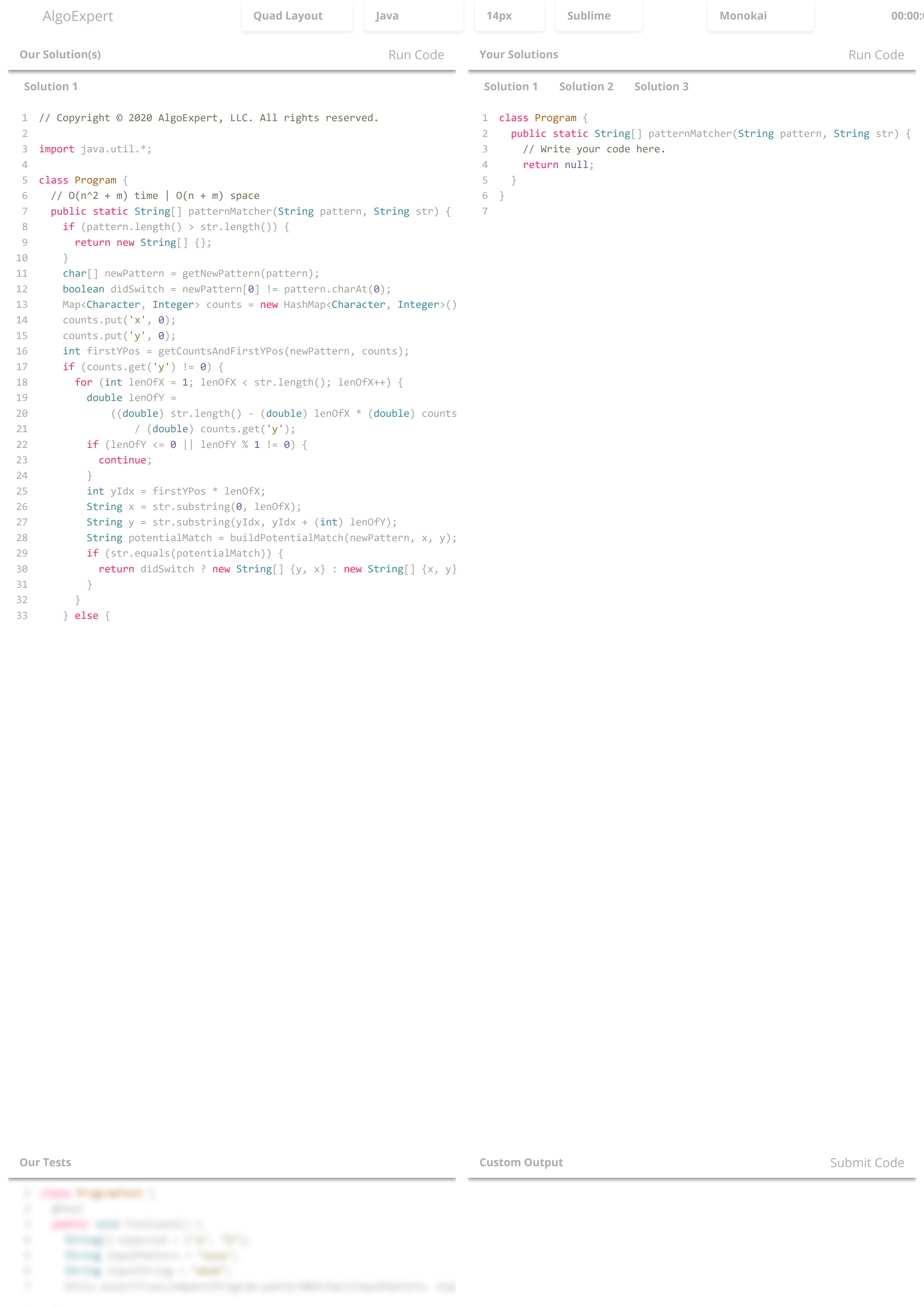

```java
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

import java.util.*;

class Program {
  // O(n^2 + m) time | O(n + m) space
  public static String[] patternMatcher(String pattern, String str) {
    if (pattern.length() > str.length()) {
      return new String[] {};
    }
    char[] newPattern = getNewPattern(pattern);
    boolean didSwitch = newPattern[0] != pattern.charAt(0);
    Map<Character, Integer> counts = new HashMap<Character, Integer>()
    counts.put('x', 0);
    counts.put('y', 0);
    int firstYPos = getCountsAndFirstYPos(newPattern, counts);
    if (counts.get('y') != 0) {
      for (int lenOfX = 1; lenOfX < str.length(); lenOfX++) {
        double lenOfY =
            ((double) str.length() - (double) lenOfX * (double) counts
                / (double) counts.get('y');
        if (lenOfY <= 0 || lenOfY % 1 != 0) {
          continue;
        }
        int yIdx = firstYPos * lenOfX;
        String x = str.substring(0, lenOfX);
        String y = str.substring(yIdx, yIdx + (int) lenOfY);
        String potentialMatch = buildPotentialMatch(newPattern, x, y);
        if (str.equals(potentialMatch)) {
          return didSwitch ? new String[] {y, x} : new String[] {x, y}
        }
      }
    } else {
```

```java
class Program {
  public static String[] patternMatcher(String pattern, String str) {
    // Write your code here.
    return null;
  }
}
```

[illegible]

```
  }

  @Test
  public void TestCase[illegible]() {
    String[] expected = {"[illegible]", "[illegible]"};
    String inputPattern = "[illegible]";
    String inputString = "[illegible]";
    Utils.assertTrue(compare(Program.patternMatcher(inputPattern, inp
  }

  @Test
  public void TestCase[illegible]() {
    String[] expected = {"[illegible]", "[illegible]"};
    String inputPattern = "[illegible]";
    String inputString = "[illegible]";
```

Run or submit code when you're ready.