```java
import static org.junit.Assert.assertEquals;

public class StringReassemblyTest {

    // Testing Combination Method

    @Test
    public void testCombinationBecause() {
        String strA = "beca";
        String strB = "cause";
        String expected = "because";
        int overlapAmt = 2;

        String combinedStr = StringReassembly.combination(strA, strB,
                overlapAmt);
        assertEquals(expected, combinedStr);
    }

    @Test
    public void testCombinationBirmingham() {
        String strA = "birming";
        String strB = "ingham";
        String expected = "birmingham";
        int overlapAmt = 3;

        String combinedStr = StringReassembly.combination(strA, strB,
                overlapAmt);
        assertEquals(expected, combinedStr);
    }

    @Test
    public void testCombinationMosquito() {
        String strA = "mosqui";
        String strB = "squito";
        String expected = "mosquito";
        int overlapAmt = 4;

        String combinedStr = StringReassembly.combination(strA, strB,
                overlapAmt);
        assertEquals(expected, combinedStr);
    }

    // Testing Add To Set Avoiding Substrings Method

    @Test
    public void testAddToSetAvoidingSubstrings_WithoutSubString() {

        Set<String> test = new Set1L<>();
        Set<String> expected = new Set1L<>();

        // add to the test set
        test.add("blue");
        test.add("red");
        test.add("green");

        // add to expected
        expected.add("blue");
        expected.add("red");
        expected.add("green");
```

```
69          expected.add("pink");
70
71          String testStr = "pink";
72
73          StringReassembly.addToSetAvoidingSubstrings(test, testStr);
74          assertEquals(expected, test);
75
76      }
77
78      @Test
79      public void testAddToSetAvoidingSubstrings_WithSubString() {
80
81          Set<String> test = new Set1L<>();
82          Set<String> expected = new Set1L<>();
83
84          // add to the test set
85          test.add("woohoo");
86          test.add("go");
87          test.add("eyes");
88
89          // add to expected
90          expected.add("woohoo");
91          expected.add("go");
92          expected.add("buckeyes");
93
94          String str = "buckeyes";
95
96          StringReassembly.addToSetAvoidingSubstrings(test, str);
97          assertEquals(expected, test);
98      }
99
100     // Testing Print With Line Separators Method
101
102     @Test
103     public void printWithLineSeparatorsTest1() {
104         SimpleWriter out = new SimpleWriter1L();
105         String testString = "Hello ~ how ~ are ~ you ~ friends?";
106
107         StringReassembly.printWithLineSeparators(testString, out);
108
109         out.close();
110     }
111
112     @Test
113     public void printWithLineSeparatorsTest2() {
114         SimpleWriter out = new SimpleWriter1L();
115         String testString = "My major is: ~Computer Science!";
116
117         StringReassembly.printWithLineSeparators(testString, out);
118
119         out.close();
120     }
121
122     @Test
123     public void printWithLineSeparatorsTest3() {
124         SimpleWriter out = new SimpleWriter1L();
125         String testString = "I a~m wri~ting~code for Project~9!";
126
127         StringReassembly.printWithLineSeparators(testString, out);
```

```
128
129         out.close();
130     }
131
132     // Testing linesFromInput
133
134     @Test
135     public void linesFromInputTestColors() {
136
137         SimpleReader in = new SimpleReader1L("data/colors");
138
139         Set<String> testSet = StringReassembly.linesFromInput(in);
140         Set<String> expected = new Set1L<>();
141
142         expected.add("colors");
143         expected.add("green");
144         expected.add("red");
145         expected.add("blue");
146
147         assertEquals(testSet, expected);
148
149     }
150
151     @Test
152     public void linesFromInputTestGreetings() {
153
154         SimpleReader in = new SimpleReader1L("data/greetings");
155
156         Set<String> testSet = StringReassembly.linesFromInput(in);
157         Set<String> expected = new Set1L<>();
158
159         expected.add("greetings");
160         expected.add("hola");
161         expected.add("bonjour");
162         expected.add("hello");
163
164         assertEquals(testSet, expected);
165
166     }
167
168 }
169
```