```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");
        expected.add("pink");

        String testStr = "pink";

        StringReassembly.addToSetAvoidingSubstrings(test, testStr);
        assertEquals(expected, test);

    }

    @Test
    public void testAddToSetAvoidingSubstrings_WithSubString() {

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

        // add to the test set
        test.add("woohoo");
        test.add("go");
        test.add("eyes");

        // add to expected
        expected.add("woohoo");
        expected.add("go");
        expected.add("buckeyes");

        String str = "buckeyes";

        StringReassembly.addToSetAvoidingSubstrings(test, str);
        assertEquals(expected, test);
    }

    // Testing Print With Line Separators Method

    @Test
    public void printWithLineSeparatorsTest1() {
        SimpleWriter out = new SimpleWriter1L();
        String testString = "Hello ~ how ~ are ~ you ~ friends?";

        StringReassembly.printWithLineSeparators(testString, out);

        out.close();
    }

    @Test
    public void printWithLineSeparatorsTest2() {
        SimpleWriter out = new SimpleWriter1L();
        String testString = "My major is: ~Computer Science!";

        StringReassembly.printWithLineSeparators(testString, out);

        out.close();
    }

    @Test
    public void printWithLineSeparatorsTest3() {
        SimpleWriter out = new SimpleWriter1L();
        String testString = "I a~m wri~ting~code for Project~9!";

        StringReassembly.printWithLineSeparators(testString, out);
```

```java

        out.close();
    }

    // Testing linesFromInput

    @Test
    public void linesFromInputTestColors() {

        SimpleReader in = new SimpleReader1L("data/colors");

        Set<String> testSet = StringReassembly.linesFromInput(in);
        Set<String> expected = new Set1L<>();

        expected.add("colors");
        expected.add("green");
        expected.add("red");
        expected.add("blue");

        assertEquals(testSet, expected);

    }

    @Test
    public void linesFromInputTestGreetings() {

        SimpleReader in = new SimpleReader1L("data/greetings");

        Set<String> testSet = StringReassembly.linesFromInput(in);
        Set<String> expected = new Set1L<>();

        expected.add("greetings");
        expected.add("hola");
        expected.add("bonjour");
        expected.add("hello");

        assertEquals(testSet, expected);

    }

}

```