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

import org.junit.Test;

import components.sequence.Sequence;
import components.sequence.Sequence1L;

/**
 * Sample JUnit test fixture for SequenceSmooth.
 *
 * @author Yunlong Zhang
 *
 */
public final class SequenceSmoothTest {

    /**
     * Constructs and returns a sequence of the integers provided as arguments.
     *
     * @param args
     *            0 or more integer arguments
     * @return the sequence of the given arguments
     * @ensures createFromArgs= [the sequence of integers in args]
     */
    private Sequence<Integer> createFromArgs(Integer... args) {
        Sequence<Integer> s = new Sequence1L<Integer>();
        for (Integer x : args) {
            s.add(s.length(), x);
        }
        return s;
    }

    /**
     * Test smooth with s1 = <2, 4, 6> and s2 = <-5, 12>.
     */
    @Test
    public void test1() {
        /*
         * Set up variables and call method under test
         */
        Sequence<Integer> seq1 = this.createFromArgs(2, 4, 6);
        Sequence<Integer> expectedSeq1 = this.createFromArgs(2, 4, 6);
        Sequence<Integer> seq2 = this.createFromArgs(-5, 12);
        Sequence<Integer> expectedSeq2 = this.createFromArgs(3, 5);
        SequenceSmooth.smooth(seq1, seq2);
        /*
         * Assert that values of variables match expectations
         */
        assertEquals(expectedSeq1, seq1);
        assertEquals(expectedSeq2, seq2);
    }

    /**
     * Test smooth with s1 = <7> and s2 = <13, 17, 11>.
     */
    @Test
    public void test2() {
        /*
         * Set up variables and call method under test
         */
```

```
60          Sequence<Integer> seq1 = this.createFromArgs(7);
61          Sequence<Integer> expectedSeq1 = this.createFromArgs(7);
62          Sequence<Integer> seq2 = this.createFromArgs(13, 17, 11);
63          Sequence<Integer> expectedSeq2 = this.createFromArgs();
64          SequenceSmooth.smooth(seq1, seq2);
65          /*
66           * Assert that values of variables match expectations
67           */
68          assertEquals(expectedSeq1, seq1);
69          assertEquals(expectedSeq2, seq2);
70      }
71
72      /**
73       * Test smooth with s1 = <7,23> and s2 = <1,2,3>.
74       */
75      @Test
76      public void test3() {
77          /*
78           * Set up variables and call method under test
79           */
80          Sequence<Integer> seq1 = this.createFromArgs(7, 23);
81          Sequence<Integer> expectedSeq1 = this.createFromArgs(7, 23);
82          Sequence<Integer> seq2 = this.createFromArgs(1, 2, 3);
83          Sequence<Integer> expectedSeq2 = this.createFromArgs(15);
84          SequenceSmooth.smooth(seq1, seq2);
85          /*
86           * Assert that values of variables match expectations
87           */
88          assertEquals(expectedSeq1, seq1);
89          assertEquals(expectedSeq2, seq2);
90      }
91
92      /**
93       * Test smooth with s1 = <7,23,2> and s2 = <1,2,3>.
94       */
95      @Test
96      public void test4() {
97          /*
98           * Set up variables and call method under test
99           */
100         Sequence<Integer> seq1 = this.createFromArgs(7, 23, 2);
101         Sequence<Integer> expectedSeq1 = this.createFromArgs(7, 23, 2);
102         Sequence<Integer> seq2 = this.createFromArgs(1, 2, 3);
103         Sequence<Integer> expectedSeq2 = this.createFromArgs(15, 12);
104         SequenceSmooth.smooth(seq1, seq2);
105         /*
106          * Assert that values of variables match expectations
107          */
108         assertEquals(expectedSeq1, seq1);
109         assertEquals(expectedSeq2, seq2);
110     }
111
112     /**
113      * Test smooth with s1 = <7,23,2,6> and s2 = <1,2,3>.
114      */
115     @Test
116     public void test5() {
117         /*
118          * Set up variables and call method under test
```

```
119          */
120         Sequence<Integer> seq1 = this.createFromArgs(7, 23, 2, 6);
121         Sequence<Integer> expectedSeq1 = this.createFromArgs(7, 23, 2, 6);
122         Sequence<Integer> seq2 = this.createFromArgs(1, 2, 3);
123         Sequence<Integer> expectedSeq2 = this.createFromArgs(15, 12, 4);
124         SequenceSmooth.smooth(seq1, seq2);
125         /*
126          * Assert that values of variables match expectations
127          */
128         assertEquals(expectedSeq1, seq1);
129         assertEquals(expectedSeq2, seq2);
130     }
131
132 }
```