```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 Put your name here
 *
 */
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
         */
        Sequence<Integer> seq1 = this.createFromArgs(7);
```

```
60          Sequence<Integer> expectedSeq1 = this.createFromArgs(7);
61          Sequence<Integer> seq2 = this.createFromArgs(13, 17, 11);
62          Sequence<Integer> expectedSeq2 = this.createFromArgs();
63          SequenceSmooth.smooth(seq1, seq2);
64          /*
65           * Assert that values of variables match expectations
66           */
67          assertEquals(expectedSeq1, seq1);
68          assertEquals(expectedSeq2, seq2);
69      }
70
71      /**
72       * Test smooth with s1 = <2, 6, 14, 18> and s2 = <17, 11>.
73       */
74      @Test
75      public void test3() {
76          /*
77           * Set up variables and call method under test
78           */
79          Sequence<Integer> seq1 = this.createFromArgs(2, 6, 14, 18);
80          Sequence<Integer> expectedSeq1 = this.createFromArgs(2, 6, 14, 18);
81          Sequence<Integer> seq2 = this.createFromArgs(17, 11);
82          Sequence<Integer> expectedSeq2 = this.createFromArgs(4, 10, 16);
83          SequenceSmooth.smooth(seq1, seq2);
84          /*
85           * Assert that values of variables match expectations
86           */
87          assertEquals(expectedSeq1, seq1);
88          assertEquals(expectedSeq2, seq2);
89      }
90
91      /**
92       * Test smooth with s1 = <4, 6, 8> and s2 = <6, 9>.
93       */
94      @Test
95      public void test4() {
96          /*
97           * Set up variables and call method under test
98           */
99          Sequence<Integer> seq1 = this.createFromArgs(4, 6, 8);
100         Sequence<Integer> expectedSeq1 = this.createFromArgs(4, 6, 8);
101         Sequence<Integer> seq2 = this.createFromArgs(6, 9);
102         Sequence<Integer> expectedSeq2 = this.createFromArgs(5, 7);
103         SequenceSmooth.smooth(seq1, seq2);
104         /*
105          * Assert that values of variables match expectations
106          */
107         assertEquals(expectedSeq1, seq1);
108         assertEquals(expectedSeq2, seq2);
109     }
110
111     /**
112      * Test smooth with s1 = <8, 12> and s2 = <4, 20>.
113      */
114     @Test
115     public void test5() {
116         /*
117          * Set up variables and call method under test
118          */
```

```
119         Sequence<Integer> seq1 = this.createFromArgs(8, 12);
120         Sequence<Integer> expectedSeq1 = this.createFromArgs(8, 12);
121         Sequence<Integer> seq2 = this.createFromArgs(4, 20);
122         Sequence<Integer> expectedSeq2 = this.createFromArgs(10);
123         SequenceSmooth.smooth(seq1, seq2);
124         /*
125          * Assert that values of variables match expectations
126          */
127         assertEquals(expectedSeq1, seq1);
128         assertEquals(expectedSeq2, seq2);
129     }
130 
131     /**
132      * Test smooth with s1 = <69> and s2 = <100>.
133      */
134     @Test
135     public void test6() {
136         /*
137          * Set up variables and call method under test
138          */
139         Sequence<Integer> seq1 = this.createFromArgs(69);
140         Sequence<Integer> expectedSeq1 = this.createFromArgs(69);
141         Sequence<Integer> seq2 = this.createFromArgs(100);
142         Sequence<Integer> expectedSeq2 = this.createFromArgs(69);
143         SequenceSmooth.smooth(seq1, seq2);
144         /*
145          * Assert that values of variables match expectations
146          */
147         assertEquals(expectedSeq1, seq1);
148         assertEquals(expectedSeq2, seq2);
149     }
150 
151 }
```