```
 1 import static org.junit.Assert.assertEquals;
 7
 8 public class NaturalNumberStaticOpsTest {
 9     //R for Routine
10     //B for Boundary
11     //C for challenging
12
13     @Test
14     //R
15     public void toStringWithCommasTest1() {
16         /*
17          * Set up variables and call method under test
18          */
19         NaturalNumber n = new NaturalNumber2(234567890);
20         NaturalNumber nCheck = new NaturalNumber2();
21         nCheck.copyFrom(n);
22         String result =
   NaturalNumberStaticOps.toStringWithCommas(n);
23         /*
24          * Assert that values of variables match
   expectations
25          */
26         assertEquals(nCheck, n); //checking for restore n
   by default
27         assertEquals(result, "234,567,890");
28     }
29
30     @Test
31     //R
32     public void toStringWithCommasTest2() {
33         /*
34          * Set up variables and call method under test
35          */
36         NaturalNumber n = new NaturalNumber2(1234);
```

```
37          NaturalNumber nCheck = new NaturalNumber2();
38          nCheck.copyFrom(n);
39          String result =
  NaturalNumberStaticOps.toStringWithCommas(n);
40          /*
41           * Assert that values of variables match
  expectations
42           */
43          assertEquals(nCheck, n); //checking for restore n
  by default
44          assertEquals(result, "1,234");
45      }
46
47      @Test
48      //R
49      public void toStringWithCommasTest21() {
50          /*
51           * Set up variables and call method under test
52           */
53          NaturalNumber n = new NaturalNumber2(1000009);
54          NaturalNumber nCheck = new NaturalNumber2();
55          nCheck.copyFrom(n);
56          String result =
  NaturalNumberStaticOps.toStringWithCommas(n);
57          /*
58           * Assert that values of variables match
  expectations
59           */
60          assertEquals(nCheck, n); //checking for restore n
  by default
61          assertEquals(result, "1,000,009");
62      }
63
64      @Test
```

```
65      //B
66      public void toStringWithCommasTest3() {
67          /*
68           * Set up variables and call method under test
69           */
70          NaturalNumber n = new NaturalNumber2(0);
71          NaturalNumber nCheck = new NaturalNumber2();
72          nCheck.copyFrom(n);
73          String result =
    NaturalNumberStaticOps.toStringWithCommas(n);
74          /*
75           * Assert that values of variables match
    expectations
76           */
77          assertEquals(nCheck, n); //checking for restore n
    by default
78          assertEquals(result, "0");
79      }
80
81      @Test
82      //B
83      public void toStringWithCommasTest31() {
84          /*
85           * Set up variables and call method under test
86           */
87          NaturalNumber n = new NaturalNumber2(1);
88          NaturalNumber nCheck = new NaturalNumber2();
89          nCheck.copyFrom(n);
90          String result =
    NaturalNumberStaticOps.toStringWithCommas(n);
91          /*
92           * Assert that values of variables match
    expectations
93           */
```

```
 94          assertEquals(nCheck, n); //checking for restore n
   by default
 95          assertEquals(result, "1");
 96      }
 97
 98      @Test
 99      //B
100      public void toStringWithCommasTest32() {
101          /*
102           * Set up variables and call method under test
103           */
104          NaturalNumber n = new NaturalNumber2(825948359);
105          NaturalNumber nCheck = new NaturalNumber2();
106          nCheck.copyFrom(n);
107          String result =
   NaturalNumberStaticOps.toStringWithCommas(n);
108          /*
109           * Assert that values of variables match
   expectations
110           */
111          assertEquals(nCheck, n); //checking for restore n
   by default
112          assertEquals(result, "825,948,359");
113      }
114
115      @Test
116      //C because there was no input for n
117      public void toStringWithCommasTest4() {
118          /*
119           * Set up variables and call method under test
120           */
121          NaturalNumber n = new NaturalNumber2();
122          NaturalNumber nCheck = new NaturalNumber2();
123          nCheck.copyFrom(n);
```

```
124         String result =
    NaturalNumberStaticOps.toStringWithCommas(n);
125         /*
126          * Assert that values of variables match
    expectations
127          */
128         assertEquals(nCheck, n); //checking for restore n
    by default
129         assertEquals(result, "0");
130     }
131
132     @Test
133     //C because zero with a negative sign
134     public void toStringWithCommasTest5() {
135         /*
136          * Set up variables and call method under test
137          */
138         NaturalNumber n = new NaturalNumber2(-0);
139         NaturalNumber nCheck = new NaturalNumber2();
140         nCheck.copyFrom(n);
141         String result =
    NaturalNumberStaticOps.toStringWithCommas(n);
142         /*
143          * Assert that values of variables match
    expectations
144          */
145         assertEquals(nCheck, n); //checking for restore n
    by default
146         assertEquals(result, "0");
147     }
148
149     @Test
150     //C because the input was not a direct number
151     public void toStringWithCommasTest6() {
```

```
152         /*
153          * Set up variables and call method under test
154          */
155         NaturalNumber n = new
    NaturalNumber2(Integer.MAX_VALUE);
156         NaturalNumber nCheck = new NaturalNumber2();
157         nCheck.copyFrom(n);
158         String result =
    NaturalNumberStaticOps.toStringWithCommas(n);
159         /*
160          * Assert that values of variables match
    expectations
161          */
162         assertEquals(nCheck, n); //checking for restore n
    by default
163         assertEquals(result, "2,147,483,647");
164     }
165 }
166
```