```java
import components.naturalnumber.NaturalNumber;

/**
 * {@code NaturalNumber} represented as a {@code String} with implementations of
 * primary methods.
 *
 * @convention <pre>
 * [all characters of $this.rep are '0' through '9']  and
 * [$this.rep does not start with '0']
 * </pre>
 * @correspondence <pre>
 * this = [if $this.rep = "" then 0
 *         else the decimal number whose ordinary depiction is $this.rep]
 * </pre>
 *
 * @author Evan Frisbie & Charan Nanduri
 *
 */
public class NaturalNumber3 extends NaturalNumberSecondary {

    /*
     * Private members
    --------------------------------------------------------
     */

    /**
     * Representation of {@code this}.
     */
    private String rep;

    /**
     * Creator of initial representation. - Done
     */
    private void createNewRep() {

        this.rep = "";

    }

    /*
     * Constructors
    ------------------------------------------------------------
     */
```

```
43
44      /**
45       * No-argument constructor. - Done
46       */
47      public NaturalNumber3() {
48          this.createNewRep();
49      }
50
51      /**
52       * Constructor from {@code int}. - Done
53       *
54       * @param i
55       *            {@code int} to initialize from
56       */
57      public NaturalNumber3(int i) {
58          assert i >= 0 : "Violation of: i >= 0";
59
60          if (i > 0) {
61              this.rep = Integer.toString(i);
62          }
63
64      }
65
66      /**
67       * Constructor from {@code String}. - Done
68       *
69       * @param s
70       *            {@code String} to initialize from
71       */
72      public NaturalNumber3(String s) {
73          assert s != null : "Violation of: s is not null";
74          assert s.matches("0|[1-9]\\d*") : ""
75                  + "Violation of: there exists n: NATURAL (s =
  TO_STRING(n))";
76
77          this.createNewRep();
78          if (Integer.parseInt(s) > 0) {
79              this.rep = s;
80          }
81
82      }
83
84      /**
85       * Constructor from {@code NaturalNumber}. - Done
86       *
```

```
 87      * @param n
 88      *            {@code NaturalNumber} to initialize from
 89      */
 90     public NaturalNumber3(NaturalNumber n) {
 91         assert n != null : "Violation of: n is not null";
 92
 93         if (n.isZero()) {
 94             this.createNewRep();
 95             this.rep = "";
 96         } else {
 97             this.createNewRep();
 98             this.rep = n.toString();
 99         }
100
101     }
102
103     /*
104      * Standard methods
    ----------------------------------------------------------
105      */
106
107     @Override
108     public final NaturalNumber newInstance() {
109         try {
110             return this.getClass().getConstructor().newInstance();
111         } catch (ReflectiveOperationException e) {
112             throw new AssertionError(
113                     "Cannot construct object of type " +
    this.getClass());
114         }
115     }
116
117     @Override
118     public final void clear() {
119         this.createNewRep();
120     }
121
122     @Override
123     public final void transferFrom(NaturalNumber source) {
124         assert source != null : "Violation of: source is not null";
125         assert source != this : "Violation of: source is not this";
126         assert source instanceof NaturalNumber3 : ""
127                 + "Violation of: source is of dynamic type
    NaturalNumberExample";
128         /*
```

```
129          * This cast cannot fail since the assert above would have
    stopped
130          * execution in that case.
131          */
132         NaturalNumber3 localSource = (NaturalNumber3) source;
133         this.rep = localSource.rep;
134         localSource.createNewRep();
135     }
136 
137     /*
138      * Kernel methods
    ------------------------------------------------------------
139      */
140 
141     //Done
142     @Override
143     public final void multiplyBy10(int k) {
144         assert 0 <= k : "Violation of: 0 <= k";
145         assert k < RADIX : "Violation of: k < 10";
146 
147         this.rep += Integer.toString(k);
148         if (this.rep.equals("0")) {
149             this.rep = "";
150         }
151 
152     }
153 
154     @Override
155     public final int divideBy10() {
156 
157         // TODO - fill in body
158         int digits = this.rep.length();
159         if (digits > 0) {
160             int remainder;
161             if (digits > 1) {
162                 remainder = Character
163                         .getNumericValue(this.rep.charAt(digits -
    1));
164                 this.rep = this.rep.substring(0, digits - 1);
165             } else {
166                 remainder =
    Character.getNumericValue(this.rep.charAt(0));
167                 this.rep = "";
168             }
169             return remainder;
```

```
170         }
171         return 0;
172     }
173 
174     @Override
175     public final boolean isZero() {
176 
177         boolean returnValue = false;
178 
179         if (this.rep.length() == 0) {
180             returnValue = true;
181         }
182 
183         return returnValue;
184     }
185 
186 }
187 
```