

```
1 import components.naturalnumber.NaturalNumber;
2 import components.naturalnumber.NaturalNumber2;
3
4 /**
5  * Extension of {@code NaturalNumber2} with secondary operations implemented as
6  * instance methods: add, subtract, and power.
7  *
8  * @author Put your name here
9  *
10 */
11 public final class NaturalNumberInstanceOps extends NaturalNumber2 {
12
13     /**
14      * No-argument constructor.
15      */
16     public NaturalNumberInstanceOps() {
17     }
18
19     /**
20      * Constructor from {@code int}.
21      *
22      * @param i
23      *      {@code int} to initialize from
24      */
25     public NaturalNumberInstanceOps(int i) {
26         super(i);
27     }
28
29     /**
30      * Constructor from {@code String}.
31      *
32      * @param s
33      *      {@code String} to initialize from
34      */
35     public NaturalNumberInstanceOps(String s) {
36         super(s);
37     }
38
39     /**
40      * Constructor from {@code NaturalNumber}.
41      *
42      * @param n
43      *      {@code NaturalNumber} to initialize from
44      */
45     public NaturalNumberInstanceOps(NaturalNumber n) {
46         super(n);
47     }
48
49     @Override
50     public void add(NaturalNumber n) {
51         assert n != null : "Violation of: n is not null";
52         /**
53          * @decreases n
54          */
55         int thisLowDigit = this.divideBy10();
56         int nLowDigit = n.divideBy10();
57         if (!n.isZero()) {
```

```
58         this.add(n);
59     }
60     thisLowDigit += nLowDigit;
61     if (thisLowDigit >= RADIX) {
62         thisLowDigit -= RADIX;
63         this.increment();
64     }
65     this.multiplyBy10(thisLowDigit);
66     n.multiplyBy10(nLowDigit);
67 }
68
69 @Override
70 public void subtract(NaturalNumber n) {
71     assert n != null : "Violation of: n is not null";
72     assert this.compareTo(n) >= 0 : "Violation of: this >= n";
73     /**
74      * @decreases n
75      */
76     int thisLowDigit = this.divideBy10();
77     int nLowDigit = n.divideBy10();
78     if (!n.isZero()) {
79         this.subtract(n);
80     }
81     thisLowDigit -= nLowDigit;
82     if (thisLowDigit < 0) {
83         thisLowDigit += RADIX;
84         this.decrement();
85     }
86     this.multiplyBy10(thisLowDigit);
87     n.multiplyBy10(nLowDigit);
88 }
89
90 @Override
91 public void power(int p) {
92     assert p >= 0 : "Violation of: p >= 0";
93
94     NaturalNumber x = new NaturalNumber2 this;
95     NaturalNumber y = new NaturalNumber2 this;
96
97     if (p % 2 == 0) {
98         x.power(p / 2);
99         x.power(2);
100     } else if (p % 2 == 1) {
101         x.power(p / 2);
102         x.power(2);
103         x.multiply(y);
104     }
105     this.copyFrom(x);
106 }
107 }
108
```