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

/**
 * @author Put your name here
 *
 */
public class CryptoUtilitiesTest {

    /*
     * Tests of reduceToGCD
     */

    @Test
    public void testReduceToGCD_0_0() {
        NaturalNumber n = new NaturalNumber2(0);
        NaturalNumber nExpected = new NaturalNumber2(0);
        NaturalNumber m = new NaturalNumber2(0);
        NaturalNumber mExpected = new NaturalNumber2(0);
        CryptoUtilities.reduceToGCD(n, m);
        assertEquals(nExpected, n);
        assertEquals(mExpected, m);
    }

    @Test
    public void testReduceToGCD_30_21() {
        NaturalNumber n = new NaturalNumber2(30);
        NaturalNumber nExpected = new NaturalNumber2(3);
        NaturalNumber m = new NaturalNumber2(21);
        NaturalNumber mExpected = new NaturalNumber2(0);
        CryptoUtilities.reduceToGCD(n, m);
        assertEquals(nExpected, n);
        assertEquals(mExpected, m);
    }

    @Test
    public void testReduceToGCD_50_200() {
        NaturalNumber n = new NaturalNumber2(50);
        NaturalNumber nExpected = new NaturalNumber2(50);
        NaturalNumber m = new NaturalNumber2(200);
        NaturalNumber mExpected = new NaturalNumber2(0);
        CryptoUtilities.reduceToGCD(n, m);
        assertEquals(nExpected, n);
        assertEquals(mExpected, m);
    }

    /*
     * Tests of isEven
     */

    @Test
    public void testIsEven_0() {
        NaturalNumber n = new NaturalNumber2(0);
        NaturalNumber nExpected = new NaturalNumber2(0);
        boolean result = CryptoUtilities.isEven(n);
        assertEquals(nExpected, n);
        assertEquals(true, result);
    }
```

```
63
64      @Test
65      public void testIsEven_1() {
66          NaturalNumber n = new NaturalNumber2(1);
67          NaturalNumber nExpected = new NaturalNumber2(1);
68          boolean result = CryptoUtilities.isEven(n);
69          assertEquals(nExpected, n);
70          assertEquals(false, result);
71      }
72
73      @Test
74      public void testIsEven_extremeCase() {
75          NaturalNumber n = new NaturalNumber2(
76                  "23947298323042039482039482092394729832 3");
77          NaturalNumber nExpected = new NaturalNumber2(
78                  "23947298323042039482039482092394729832 3");
79          boolean result = CryptoUtilities.isEven(n);
80          assertEquals(nExpected, n);
81          assertEquals(false, result);
82      }
83
84      @Test
85      public void testIsEven_normalCase() {
86          NaturalNumber n = new NaturalNumber2("68");
87          NaturalNumber nExpected = new NaturalNumber2("68");
88          boolean result = CryptoUtilities.isEven(n);
89          assertEquals(nExpected, n);
90          assertEquals(true, result);
91      }
92
93      /*
94       * Tests of powerMod
95       */
96
97      @Test
98      public void testPowerMod_0_0_2() {
99          NaturalNumber n = new NaturalNumber2(0);
100         NaturalNumber nExpected = new NaturalNumber2(1);
101         NaturalNumber p = new NaturalNumber2(0);
102         NaturalNumber pExpected = new NaturalNumber2(0);
103         NaturalNumber m = new NaturalNumber2(2);
104         NaturalNumber mExpected = new NaturalNumber2(2);
105         CryptoUtilities.powerMod(n, p, m);
106         assertEquals(nExpected, n);
107         assertEquals(pExpected, p);
108         assertEquals(mExpected, m);
109     }
110
111     @Test
112     public void testPowerMod_17_18_19() {
113         NaturalNumber n = new NaturalNumber2(17);
114         NaturalNumber nExpected = new NaturalNumber2(1);
115         NaturalNumber p = new NaturalNumber2(18);
116         NaturalNumber pExpected = new NaturalNumber2(18);
117         NaturalNumber m = new NaturalNumber2(19);
118         NaturalNumber mExpected = new NaturalNumber2(19);
119         CryptoUtilities.powerMod(n, p, m);
```

```
120         assertEquals(nExpected, n);
121         assertEquals(pExpected, p);
122         assertEquals(mExpected, m);
123     }
124 
125     @Test
126     public void testPowerMod_extremeCase() {
127         NaturalNumber n = new NaturalNumber2(100);
128         NaturalNumber nExpected = new NaturalNumber2(1);
129         NaturalNumber p = new NaturalNumber2(67);
130         NaturalNumber pExpected = new NaturalNumber2(67);
131         NaturalNumber m = new NaturalNumber2(9);
132         NaturalNumber mExpected = new NaturalNumber2(9);
133         CryptoUtilities.powerMod(n, p, m);
134         assertEquals(nExpected, n);
135         assertEquals(pExpected, p);
136         assertEquals(mExpected, m);
137     }
138 
139     @Test
140     public void testIsWitnessToCompositeness1() {
141         NaturalNumber n = new NaturalNumber2(5);
142         NaturalNumber nExpected = new NaturalNumber2(5);
143         NaturalNumber w = new NaturalNumber2(3);
144         NaturalNumber wExpected = new NaturalNumber2(3);
145         boolean expectedAnswer = false;
146         assertEquals(CryptoUtilities.isWitnessToCompositeness(w, n),
147                 expectedAnswer);
148     }
149 
150     @Test
151     public void testIsWitnessToCompositeness2() {
152         NaturalNumber n = new NaturalNumber2(20);
153         NaturalNumber w = new NaturalNumber2(4);
154         boolean expectedAnswer = true;
155         assertEquals(CryptoUtilities.isWitnessToCompositeness(w, n),
156                 expectedAnswer);
157     }
158 
159     @Test
160     public void testIsWitnessToCompositeness3() {
161         NaturalNumber n = new NaturalNumber2(23);
162         NaturalNumber w = new NaturalNumber2(15);
163         boolean expectedAnswer = false;
164         assertEquals(CryptoUtilities.isWitnessToCompositeness(w, n),
165                 expectedAnswer);
166     }
167 
168     @Test
169     public void testIsPrime2_1() {
170         NaturalNumber n = new NaturalNumber2(100);
171         boolean expectedAnswer = false;
172         assertEquals(CryptoUtilities.isPrime2(n), expectedAnswer);
173     }
174 
175     @Test
176     public void testIsPrime2_2() {
```

```
177         NaturalNumber n = new NaturalNumber2(5);
178         boolean expectedAnswer = true;
179         assertEquals(CryptoUtilities.isPrime2(n), expectedAnswer);
180     }
181 
182     @Test
183     public void testIsPrime2_3() {
184         NaturalNumber n = new NaturalNumber2(26);
185         boolean expectedAnswer = false;
186         assertEquals(CryptoUtilities.isPrime2(n), expectedAnswer);
187     }
188 
189     @Test
190     public void testGenerateNextLikelyPrime_1() {
191         NaturalNumber n = new NaturalNumber2(5);
192         NaturalNumber expectedAnswer = new NaturalNumber2(7);
193         CryptoUtilities.generateNextLikelyPrime(n);
194         assertEquals(n, expectedAnswer);
195     }
196 
197     @Test
198     public void testGenerateNextLikelyPrime_2() {
199         NaturalNumber n = new NaturalNumber2(29);
200         NaturalNumber expectedAnswer = new NaturalNumber2(31);
201         CryptoUtilities.generateNextLikelyPrime(n);
202         assertEquals(n, expectedAnswer);
203     }
204 
205     @Test
206     public void testGenerateNextLikelyPrime_3() {
207         NaturalNumber n = new NaturalNumber2(137);
208         NaturalNumber expectedAnswer = new NaturalNumber2(139);
209         CryptoUtilities.generateNextLikelyPrime(n);
210         assertEquals(n, expectedAnswer);
211     }
212 
213 }
214 
```