

```
1 package com.ffcg;
2
3 import org.junit.Test;
4
5 public class DivisorTest {
6
7     @Test
8     public void
9     return_false_for_0_as_divisor_of_any_number(){
10         assert(!(new Divisor(0)).isDivisorOf(0));
11         assert(!(new Divisor(0)).isDivisorOf(1));
12         assert(!(new Divisor(0)).isDivisorOf(2));
13         assert(!(new Divisor(0)).isDivisorOf(3));
14         assert(!(new Divisor(0)).isDivisorOf(4));
15     }
16
17     @Test
18     public void return_false_for_2_as_divisor_of_0
19     () {
20         assert(!(new Divisor(2)).isDivisorOf(0));
21     }
22
23     @Test
24     public void return_false_for_2_as_divisor_of_1
25     () {
26         assert(!(new Divisor(2)).isDivisorOf(1));
27     }
28
29     @Test
30     public void return_false_for_2_as_divisor_of_3
31     () {
32         assert(!(new Divisor(2)).isDivisorOf(3));
33     }
34
35     @Test
36     public void return_false_for_4_as_divisor_of_91
37     () {
38         assert(!(new Divisor(4)).isDivisorOf(91));
39     }
40 }
```

```
37     @Test
38     public void return_true_for_2_as_divisor_of_4() {
39         assert((new Divisor(2)).isDivisorOf(4));
40     }
41
42     @Test
43     public void return_true_for_3_as_divisor_of_21
44     () {
45         assert((new Divisor(3)).isDivisorOf(21));
46     }
47
48     @Test
49     public void return_true_for_2_as_divisor_of_2() {
50         assert((new Divisor(2)).isDivisorOf(2));
51     }
```

```
1 package com.ffcg;
2
3 import org.junit.Test;
4
5 import java.util.ArrayList;
6 import java.util.List;
7
8
9 public class SetDivisorsTest {
10     @Test
11     public void
12     return_true_when_calculating_all_divisors_of_25() {
13         List<Divisor> listDivisorsOf25 = new
14         ArrayList<Divisor>();
15         listDivisorsOf25.add(new Divisor(1));
16         listDivisorsOf25.add(new Divisor(5));
17         listDivisorsOf25.add(new Divisor(25));
18
19         SetDivisors listDivisors = new SetDivisors(25
20 );
21
22         assert(listDivisors.isEqualTo(
23 listDivisorsOf25));
24
25     }
26
27     @Test
28     public void
29     return_true_when_calculating_all_divisors_of_23() {
30         List<Divisor> listDivisorsOf23 = new
31         ArrayList<Divisor>();
32         listDivisorsOf23.add(new Divisor(1));
33         listDivisorsOf23.add(new Divisor(23));
34
35         SetDivisors listDivisors = new SetDivisors(23
36 );
37
38         assert(listDivisors.isEqualTo(
39 listDivisorsOf23));
40
41     }
42 }
```

```
34
35     @Test
36     public void
    return_true_when_calculating_sum_of_squared_divisors_
    of_25() {
37         SetDivisors listDivisorsOf25 = new
    SetDivisors(25);
38
39         assert((1*1 + 5*5 + 25*25) ==
    listDivisorsOf25.getSumOfSquaredDivisors());
40     }
41
42     @Test
43     public void
    return_true_when_calculating_sum_of_squared_divisors_
    of_23(){
44         SetDivisors listDivisorsOf23 = new
    SetDivisors(23);
45         assert((1*1 + 23*23) == listDivisorsOf23.
    getSumOfSquaredDivisors());
46
47     }
48 }
49
```

```
1 package com.ffcg;
2
3 import org.junit.Test;
4
5
6 public class PerfectSquaredTest {
7     @Test
8     public void
9     return_true_for_numbers_are_perfect_squared() {
10         assert(PerfectSquared.isPerfectSquared(1));
11         assert(PerfectSquared.isPerfectSquared(4));
12         assert(PerfectSquared.isPerfectSquared(9));
13         assert(PerfectSquared.isPerfectSquared(16));
14         assert(PerfectSquared.isPerfectSquared(25));
15     }
16     @Test
17     public void
18     return_false_for_numbers_are_not_perfect_squared() {
19         assert(!PerfectSquared.isPerfectSquared(2));
20         assert(!PerfectSquared.isPerfectSquared(3));
21         assert(!PerfectSquared.isPerfectSquared(5));
22         assert(!PerfectSquared.isPerfectSquared(6));
23         assert(!PerfectSquared.isPerfectSquared(7));
24     }
25 }
```

```
1 package com.ffcg;
2
3 import org.junit.Test;
4
5 import java.util.Arrays;
6
7 public class SquaredDivisorTest {
8
9     @Test
10     public void
11     return_true_calculating_all_squared_divisors_between_
12     1_and_250() {
13         int [] squaredDivisorsBetween1And250 = {1, 42
14         , 246};
15
16         int [] squaredDivisors = SquaredDivisor.
17         searchSquaredDivisor(1, 250);
18
19         assert(Arrays.equals(
20         squaredDivisorsBetween1And250, squaredDivisors));
21     }
22
23     @Test
24     public void
25     return_true_calculating_all_squared_divisors_between_
26     42_and_250() {
27         int [] squaredDivisorsBetween1And250 = {42,
28         246};
29
30         int [] squaredDivisors = SquaredDivisor.
31         searchSquaredDivisor(42, 250);
32
33         assert(Arrays.equals(
34         squaredDivisorsBetween1And250, squaredDivisors));
35     }
36 }
```