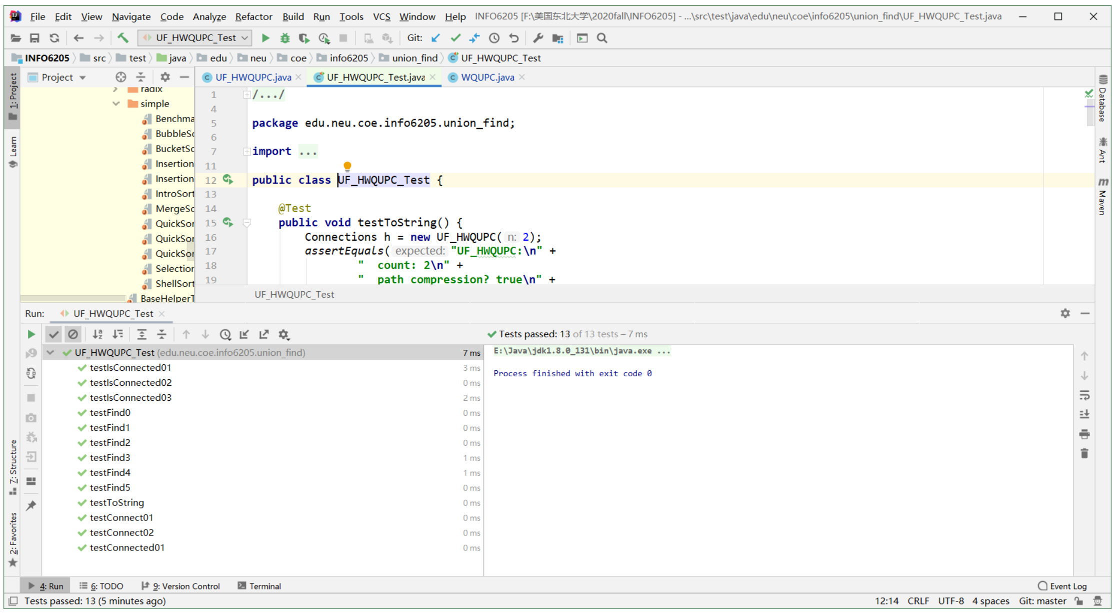
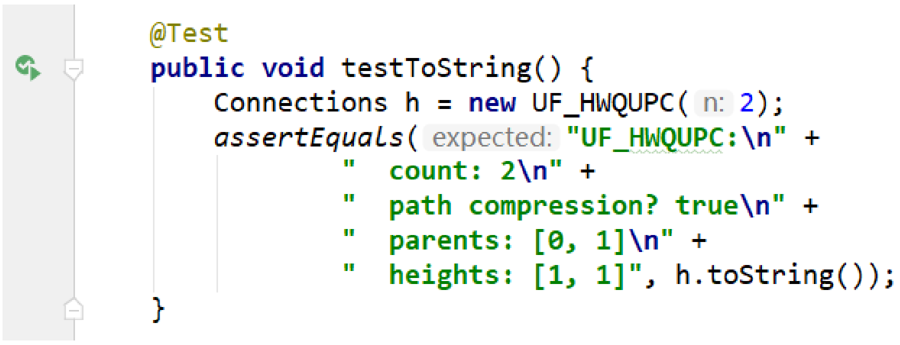


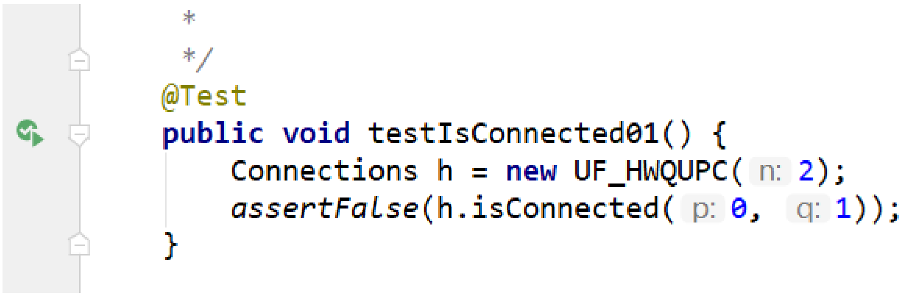
```
INFO6205 [F:\美国东北大学\2020fall\INFO6205] - ...\src\test\java\edu\neu\coe\info6205\union_find\UF_HWQUPC_Test.java

/.../

package edu.neu.coe.info6205.union_find;

import ...

public class UF_HWQUPC_Test {

    @Test
    public void testToString() {
        Connections h = new UF_HWQUPC( n: 2);
        assertEquals( expected: "UF_HWQUPC:\n" +
                "  count: 2\n" +
                "  path compression? true\n" +
```

```
Tests passed: 13 of 13 tests – 7 ms
UF_HWQUPC_Test (edu.neu.coe.info6205.union_find)   7 ms
  testIsConnected01   3 ms
  testIsConnected02   0 ms
  testIsConnected03   2 ms
  testFind0           0 ms
  testFind1           0 ms
  testFind2           0 ms
  testFind3           1 ms
  testFind4           1 ms
  testFind5           0 ms
  testToString        0 ms
  testConnect01       0 ms
  testConnect02       0 ms
  testConnected01     0 ms

E:\Java\jdk1.8.0_131\bin\java.exe ...

Process finished with exit code 0
```

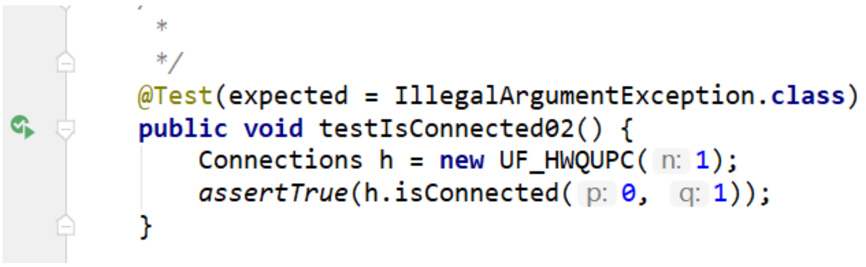
```
Tests passed: 13 (5 minutes ago)
```



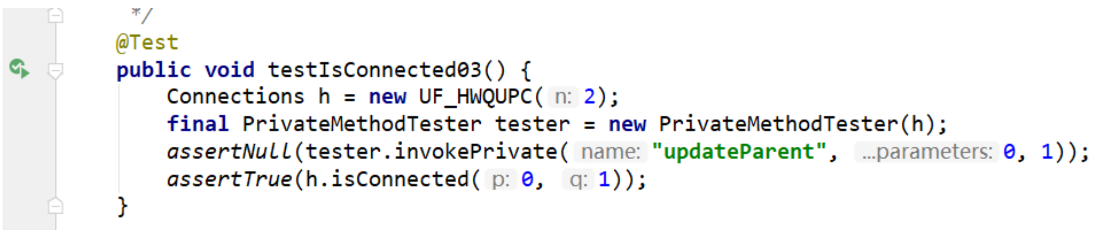
```
    @Test
    public void testToString() {
        Connections h = new UF_HWQUPC( n: 2);
        assertEquals( expected: "UF_HWQUPC:\n" +
                "  count: 2\n" +
                "  path compression? true\n" +
                "  parents: [0, 1]\n" +
                "  heights: [1, 1]", h.toString());
    }
```



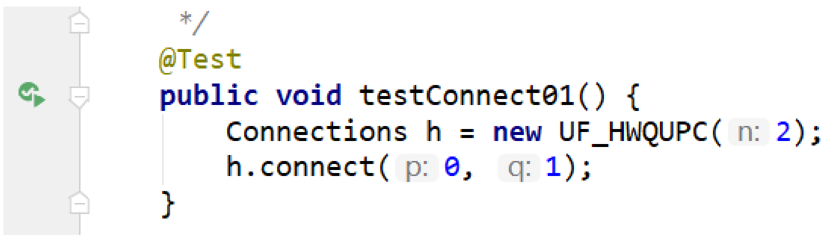
```
     *
     */
    @Test
    public void testIsConnected01() {
        Connections h = new UF_HWQUPC( n: 2);
        assertFalse(h.isConnected( p: 0,  q: 1));
    }
```



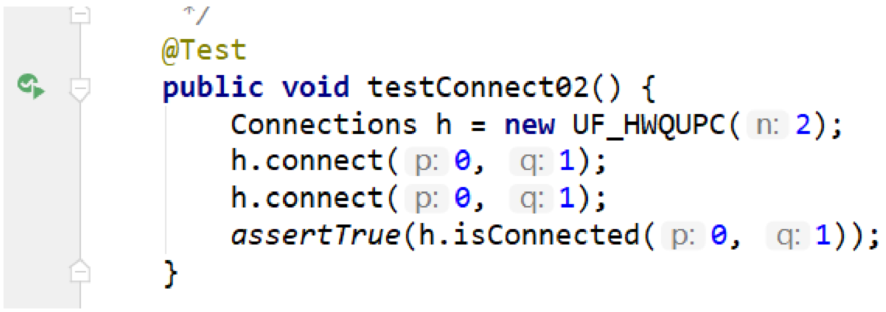
```
     *
     */
    @Test(expected = IllegalArgumentException.class)
    public void testIsConnected02() {
        Connections h = new UF_HWQUPC( n: 1);
        assertTrue(h.isConnected( p: 0,  q: 1));
    }
```



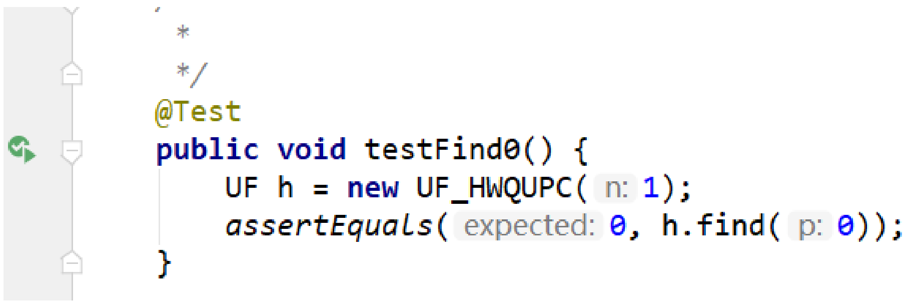
```
     */
    @Test
    public void testIsConnected03() {
        Connections h = new UF_HWQUPC( n: 2);
        final PrivateMethodTester tester = new PrivateMethodTester(h);
        assertNull(tester.invokePrivate( name: "updateParent",  ...parameters: 0, 1));
        assertTrue(h.isConnected( p: 0,  q: 1));
    }
```



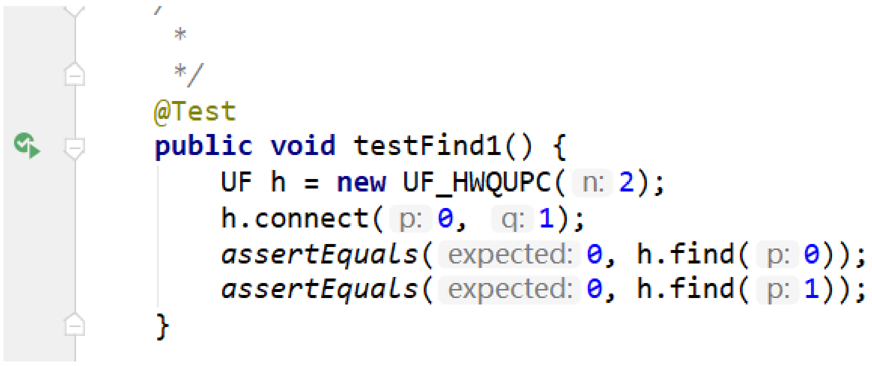
```java
     */
    @Test
    public void testConnect01() {
        Connections h = new UF_HWQUPC( n: 2);
        h.connect( p: 0,  q: 1);
    }
```



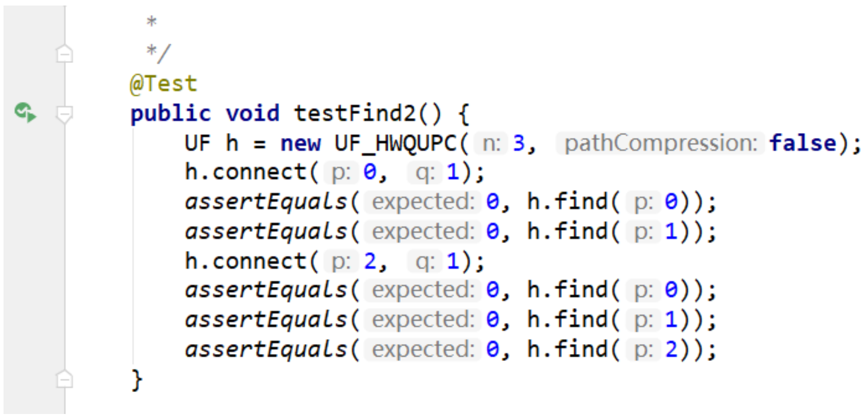
```java
     */
    @Test
    public void testConnect02() {
        Connections h = new UF_HWQUPC( n: 2);
        h.connect( p: 0,  q: 1);
        h.connect( p: 0,  q: 1);
        assertTrue(h.isConnected( p: 0,  q: 1));
    }
```



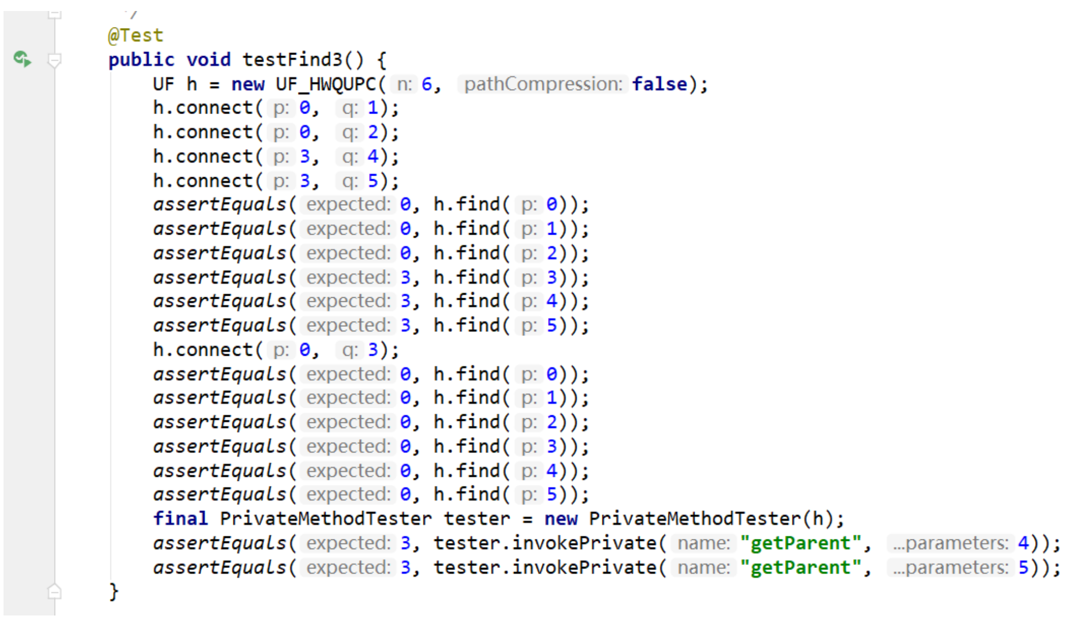
```java
    ,
     *
     */
    @Test
    public void testFind0() {
        UF h = new UF_HWQUPC( n: 1);
        assertEquals( expected: 0, h.find( p: 0));
    }
```



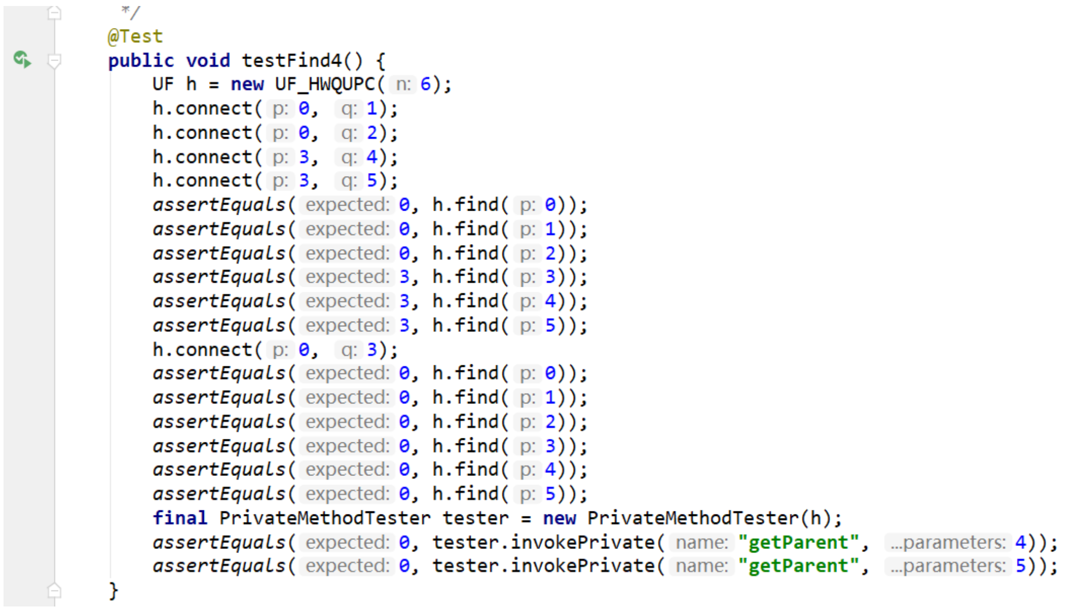
```java
    ,
     *
     */
    @Test
    public void testFind1() {
        UF h = new UF_HWQUPC( n: 2);
        h.connect( p: 0,  q: 1);
        assertEquals( expected: 0, h.find( p: 0));
        assertEquals( expected: 0, h.find( p: 1));
    }
```



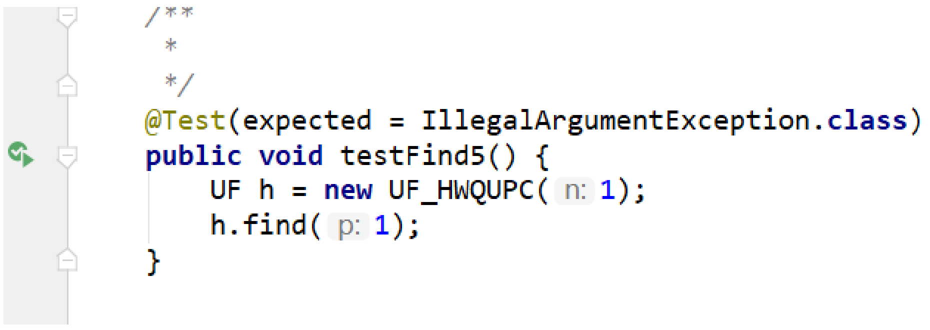
```java
     *
     */
    @Test
    public void testFind2() {
        UF h = new UF_HWQUPC( n: 3,  pathCompression: false);
        h.connect( p: 0,  q: 1);
        assertEquals( expected: 0, h.find( p: 0));
        assertEquals( expected: 0, h.find( p: 1));
        h.connect( p: 2,  q: 1);
        assertEquals( expected: 0, h.find( p: 0));
        assertEquals( expected: 0, h.find( p: 1));
        assertEquals( expected: 0, h.find( p: 2));
    }
```



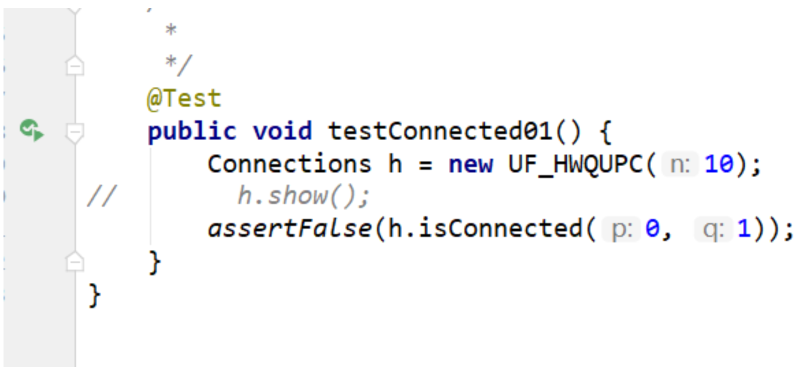
```
 */
@Test
public void testFind3() {
    UF h = new UF_HWQUPC( n: 6,  pathCompression: false);
    h.connect( p: 0,  q: 1);
    h.connect( p: 0,  q: 2);
    h.connect( p: 3,  q: 4);
    h.connect( p: 3,  q: 5);
    assertEquals( expected: 0, h.find( p: 0));
    assertEquals( expected: 0, h.find( p: 1));
    assertEquals( expected: 0, h.find( p: 2));
    assertEquals( expected: 3, h.find( p: 3));
    assertEquals( expected: 3, h.find( p: 4));
    assertEquals( expected: 3, h.find( p: 5));
    h.connect( p: 0,  q: 3);
    assertEquals( expected: 0, h.find( p: 0));
    assertEquals( expected: 0, h.find( p: 1));
    assertEquals( expected: 0, h.find( p: 2));
    assertEquals( expected: 0, h.find( p: 3));
    assertEquals( expected: 0, h.find( p: 4));
    assertEquals( expected: 0, h.find( p: 5));
    final PrivateMethodTester tester = new PrivateMethodTester(h);
    assertEquals( expected: 3, tester.invokePrivate( name: "getParent",  ...parameters: 4));
    assertEquals( expected: 3, tester.invokePrivate( name: "getParent",  ...parameters: 5));
}
```



```
 */
@Test
public void testFind4() {
    UF h = new UF_HWQUPC( n: 6);
    h.connect( p: 0,  q: 1);
    h.connect( p: 0,  q: 2);
    h.connect( p: 3,  q: 4);
    h.connect( p: 3,  q: 5);
    assertEquals( expected: 0, h.find( p: 0));
    assertEquals( expected: 0, h.find( p: 1));
    assertEquals( expected: 0, h.find( p: 2));
    assertEquals( expected: 3, h.find( p: 3));
    assertEquals( expected: 3, h.find( p: 4));
    assertEquals( expected: 3, h.find( p: 5));
    h.connect( p: 0,  q: 3);
    assertEquals( expected: 0, h.find( p: 0));
    assertEquals( expected: 0, h.find( p: 1));
    assertEquals( expected: 0, h.find( p: 2));
    assertEquals( expected: 0, h.find( p: 3));
    assertEquals( expected: 0, h.find( p: 4));
    assertEquals( expected: 0, h.find( p: 5));
    final PrivateMethodTester tester = new PrivateMethodTester(h);
    assertEquals( expected: 0, tester.invokePrivate( name: "getParent",  ...parameters: 4));
    assertEquals( expected: 0, tester.invokePrivate( name: "getParent",  ...parameters: 5));
}
```



```
/**
 *
 */
@Test(expected = IllegalArgumentException.class)
public void testFind5() {
    UF h = new UF_HWQUPC( n: 1);
    h.find( p: 1);
}
```



```
     *
     */
    @Test
    public void testConnected01() {
        Connections h = new UF_HWQUPC( n: 10);
//          h.show();
        assertFalse(h.isConnected( p: 0,  q: 1));
    }
}
```