软件测试上机报告



第二次上机作业 –Junit and Jacoco

**学 院 智能与计算学部**

**专    业 软件工程**

**姓 名 白文杰**

**学 号 3018216031**

**年 级 2018**

**班    级 1**

#### Experimental requirements

1. Install Junit(4.12), Hamcrest(1.3), Eclemma or Jacoco(newest) in Eclipse (done by last lab) or IDEA (follow the new instruction).
2. Write a java program for the given problem and test the program with **Advanced Junit Usage**.
3. Add `@Before`, `@BeforeClass`, `@After`, `@AfterClass` to different functions, and write logs in each function, observe the execution order.
4. Given the input list `testinput.txt`, each line with a test input and expected output, separated by comma, eg. `10,36` means `10 = 3 + 3 + 4` and the result should be `3 \* 3 \* 4 = 36`. Each line only contains one test case. Use `@RunWith(Parameterized.class)` to load the test file, and test all test cases.
5. Use Eclemma or Jacoco to produce coverage.
6. Description of the problem:

Given a positive integer N, split it into the sum of at least two positive integers, and maximize the product of these integers. Returns the largest product value M.

**Input:**

**N** a positive integer

**Output:**

**M** the largest product value

Sample Input:

10

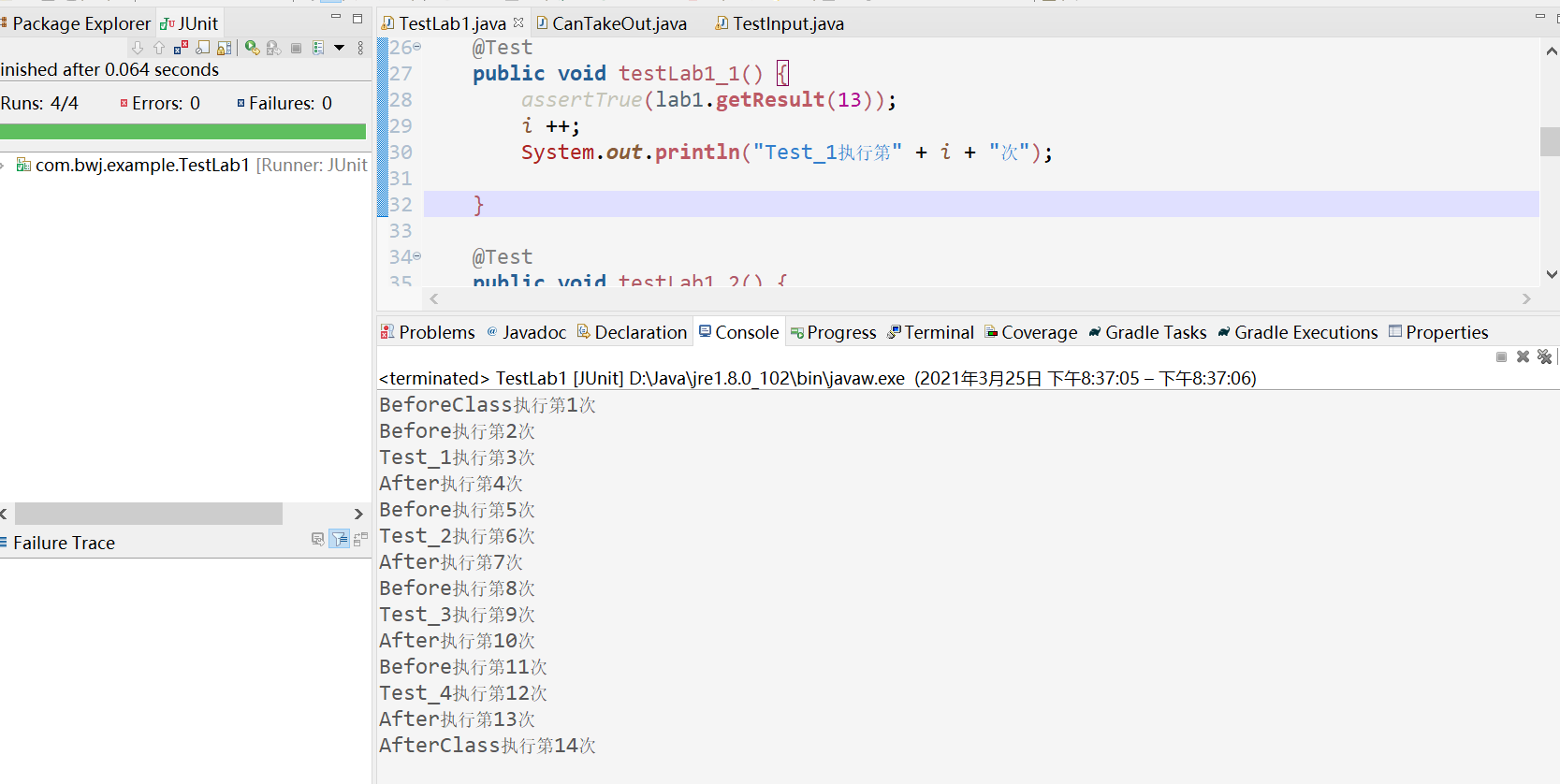
Sample Output

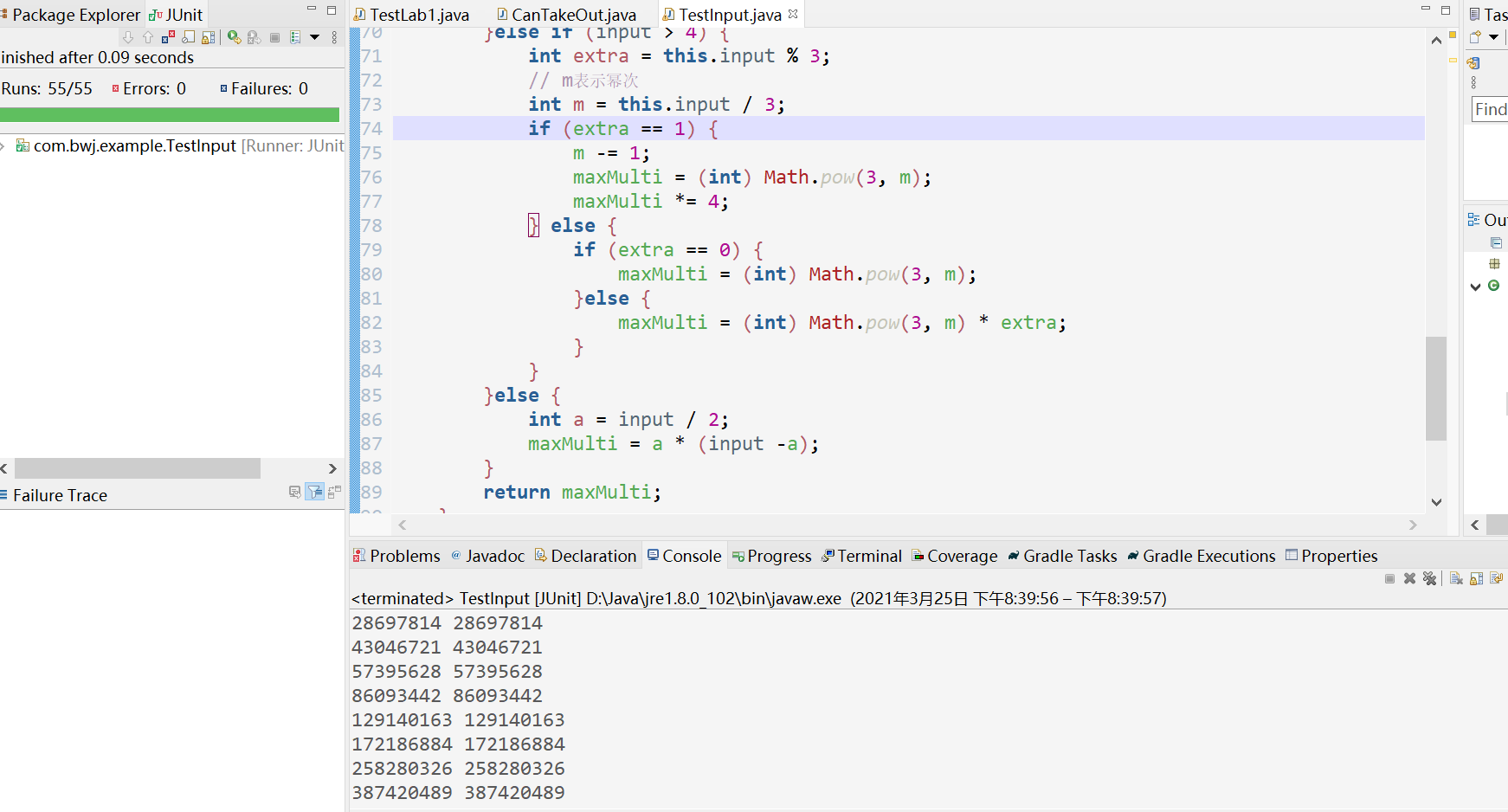
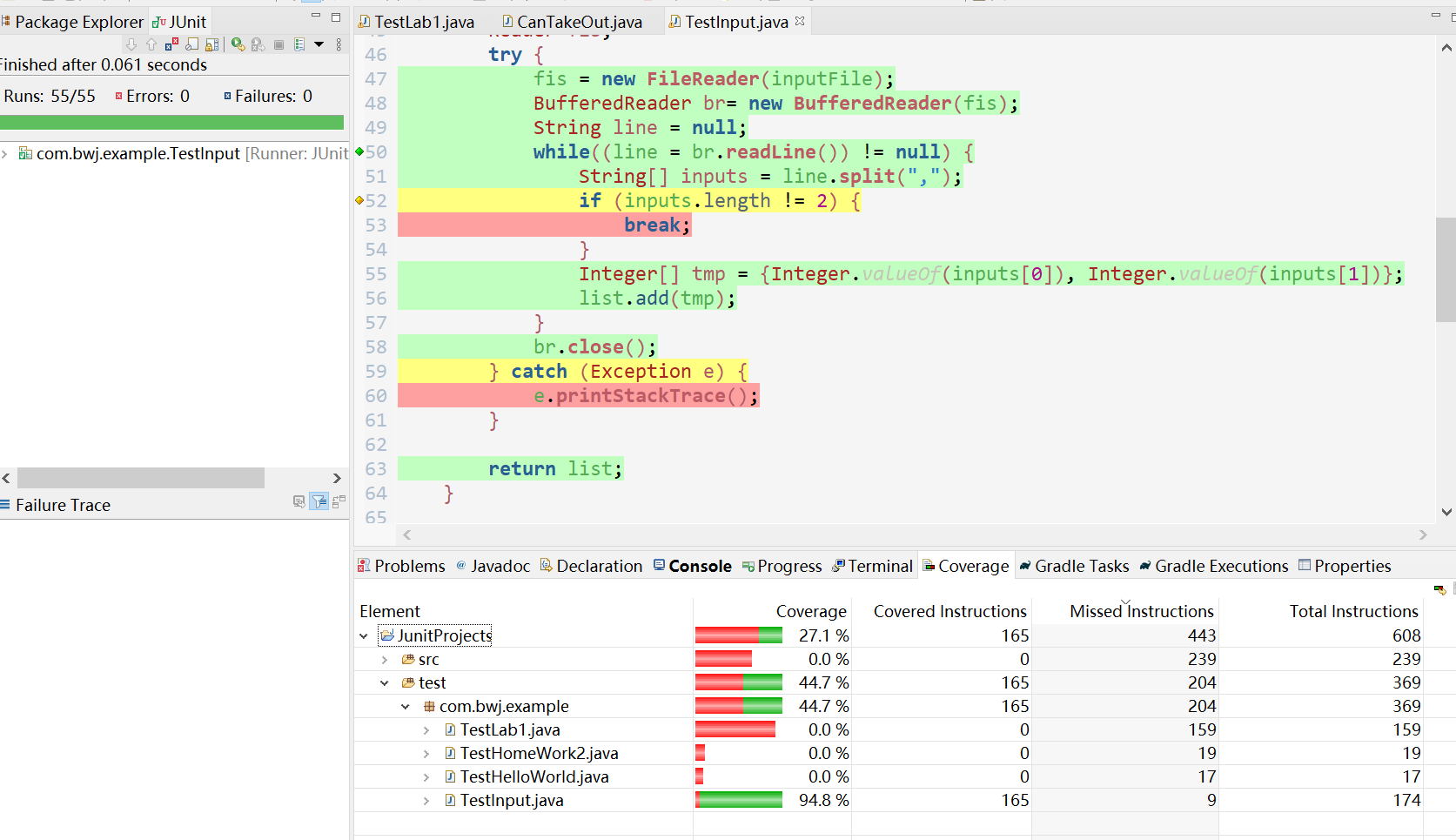
36 (10 = 3 + 4 + 4， 3 \*3\*4 = 36)

#### Configuration

1. 对于要求4，我仍然使用了实验1的代码但是略有修改。详细的代码可以参考Part 4中的TestLab1.java。

执行的结果是：



1. 对于要求5，可以参考Part4中的TestInput.java文件。最后的执行结果是：  
   
2. 使用Eclemma来查看代码的覆盖率：  
   

#### Result analysis

1. 要求4中，几个注解主要的区别是：  
   @BeforeClass：在测试的最开始执行  
   @AfterClass：在测试的最后执行  
   @Before：在每个测试前运行  
   @After：在每个测试后运行  
   @Test：说明此段代码需要进行测试  
   对于整段程序的执行顺序为：@BeforeClass -> @Before -> @Test -> @After -> @AfterClass  
   对于一个测试来说的执行顺序为：@Before -> @Test -> @After
2. 要求5中，@RunWith(Parameterized.class)这一注解的作用是要是将测试用例集整体进行测试。在这里遇到一个小坑是函数得到返回值没有返回正确，最后才发现必须是List<Integeter[]>才可以。  
   同时我还在给@Parameters注解添加了参数：(name = "{index}: maxMutli({0})={1}")，作用是让最后的测试结果易于区分。
3. 可以看到最后的代码覆盖率很高，有94.8%。也就是说几乎所有的代码都执行到了，没有执行的部分只是一些异常处理部分。

#### Source code

CanTakeOut.java

|  |
| --- |
| package com**.**bwj**.**example**;**  public class CanTakeOut **{**  final int numOf50 **=** 1**;**  final int numOf20 **=** 1**;**  final int numOf10 **=** 1**;**  final int numOf5 **=** 2**;**  final int numOf1 **=** 3**;**    int remain50**;**  int remain20**;**  int remain10**;**  int remain5**;**  int remain1**;**    int x**;**    public CanTakeOut**()** **{**  reset**();**  **}**  private boolean canTakeOut**(**int x**)** **{**  **if** **(**x **>=** 50 **&&** remain50 **>** 0**)** **{**  x **-=** 50**;**  remain50 **--;**  **if** **(**x **==** 0**)** **{**  **return** **true;**  **}**  **}**    **if** **(**x **>=** 20 **&&** remain20 **>** 0**)** **{**  x **-=** 20**;**  remain20 **--;**  **if** **(**x **==** 0**)** **{**  **return** **true;**  **}**  **}**    **if** **(**x **>=** 10 **&&** remain10 **>** 0**)** **{**  x **-=** 10**;**  remain10 **--;**  **if** **(**x **==** 0**)** **{**  **return** **true;**  **}**  **}**    **if** **(**x **>=** 5 **&&** remain5 **>** 0**)** **{**  x **-=** 5**;**  remain5 **--;**  **if** **(**x **==** 0**)** **{**  **return** **true;**  **}**  **}**  **if** **(**x **>=** 5**)** **{**  **if** **(**remain1 **>** 0 **||** remain10 **>** 0 **||** remain20 **>** 0 **||** remain5 **>** 0 **||** remain50 **>** 0**)** **{**  canTakeOut**(**x**);**  **}**  **}else** **if** **(**x **>** remain1**)** **{**  **return** **false;**  **}else** **{**  **return** **true;**  **}**      **return** **false;**  **}**    public boolean getResult**(**int a**)** **{**  **this.**x **=** a**;**  int maxValue **=** 50 **\*** numOf50 **+** 20 **\*** numOf20 **+** 10 **\*** numOf10 **+** 5 **\*** numOf5 **+** numOf1**;**  int minValue **=** 0**;**  **if** **(**x **>** maxValue **||** x **<** minValue**)** **{**  **return** **false;**  **}**  boolean tmp **=** canTakeOut**(this.**x**);**  reset**();**  **return** tmp**;**  **}**    public void reset**()** **{**  **this.**remain50 **=** **this.**numOf50**;**  **this.**remain20 **=** **this.**numOf20**;**  **this.**remain10 **=** **this.**numOf10**;**  **this.**remain5 **=** **this.**numOf5**;**  **this.**remain1 **=** **this.**numOf1**;**  **}**  **}** |

TestLab1.java

|  |
| --- |
| package com**.**bwj**.**example**;**  **import** static org**.**junit**.**Assert**.\*;**  **import** org**.**junit**.\*;**  **import** junit**.**extensions**.**ActiveTestSuite**;**  public class TestLab1 **{**  public CanTakeOut lab1**;**  public static int i **=** 0**;**    @BeforeClass  public static void beforeClass**()** **{**  i **++;**  System**.**out**.**println**(**"BeforeClass执行第" **+** i **+** "次"**);**  **}**    @Before  public void init**()** **{**  lab1 **=** **new** CanTakeOut**();**  i **++;**  System**.**out**.**println**(**"Before执行第" **+** i **+** "次"**);**  **}**    @Test  public void testLab1\_1**()** **{**  assertTrue**(**lab1**.**getResult**(**13**));**  i **++;**  System**.**out**.**println**(**"Test\_1执行第" **+** i **+** "次"**);**    **}**    @Test  public void testLab1\_2**()** **{**  assertTrue**(**lab1**.**getResult**(**23**));**  i **++;**  System**.**out**.**println**(**"Test\_2执行第" **+** i **+** "次"**);**    **}**    @Test  public void testLab1\_3**()** **{**  assertFalse**(**lab1**.**getResult**(**39**));**  i **++;**  System**.**out**.**println**(**"Test\_3执行第" **+** i **+** "次"**);**    **}**    @Test  public void testLab1\_4**()** **{**  assertFalse**(**lab1**.**getResult**(**200**));**  i **++;**  System**.**out**.**println**(**"Test\_4执行第" **+** i **+** "次"**);**  **}**    @After  public void release**()** **{**  i **++;**  System**.**out**.**println**(**"After执行第" **+** i **+** "次"**);**  **}**    @AfterClass  public static void afterClass**()** **{**  i **++;**  System**.**out**.**println**(**"AfterClass执行第" **+** i **+** "次"**);**  **}**  **}** |

TestInput.java

|  |
| --- |
| package com**.**bwj**.**example**;**  **import** java**.**util**.**ArrayList**;**  **import** java**.**util**.**Arrays**;**  **import** java**.**util**.**List**;**  **import** static org**.**junit**.**Assert**.\*;**  **import** java**.**io**.**BufferedReader**;**  **import** java**.**io**.**File**;**  **import** java**.**io**.**FileReader**;**  **import** java**.**io**.**Reader**;**  **import** org**.**junit**.**Test**;**  **import** org**.**junit**.**runner**.**RunWith**;**  **import** org**.**junit**.**runners**.**Parameterized**;**  **import** org**.**junit**.**runners**.**Parameterized**.**Parameters**;**  @RunWith**(**Parameterized**.**class**)**  public class TestInput **{**  public int input**;**  public int expected**;**    public TestInput**(**int input**,** int expected**)** **{**  **this.**input **=** input**;**  **this.**expected **=** expected**;**  **}**    @Parameters**(**name **=** "{index}: maxMutli({0})={1}"**)**  public static List**<**Integer**[]>** parameters**(){**  List**<**Integer**[]>** list **=** **new** ArrayList**<>();**  File inputFile **=** **new** File**(**"testinput.txt"**);**  **if** **(!**inputFile**.**exists**())** **{**  System**.**out**.**println**(**"未找到文件"**);**  System**.**exit**(-**1**);**  **}**  Reader fis**;**  **try** **{**  fis **=** **new** FileReader**(**inputFile**);**  BufferedReader br**=** **new** BufferedReader**(**fis**);**  String line **=** **null;**  **while((**line **=** br**.**readLine**())** **!=** **null)** **{**  String**[]** inputs **=** line**.**split**(**","**);**  **if** **(**inputs**.**length **!=** 2**)** **{**  **break;**  **}**  Integer**[]** tmp **=** **{**Integer**.**valueOf**(**inputs**[**0**]),** Integer**.**valueOf**(**inputs**[**1**])};**  list**.**add**(**tmp**);**  **}**  br**.**close**();**  **}** **catch** **(**Exception e**)** **{**  e**.**printStackTrace**();**  **}**    **return** list**;**  **}**    public int maxMutli**()** **{**  int maxMulti **=** 0**;**  **if** **(**input **==** 1**)** **{**  maxMulti **=** 1**;**  **}else** **if** **(**input **>** 4**)** **{**  int extra **=** **this.**input **%** 3**;**  // m表示幂次  int m **=** **this.**input **/** 3**;**  **if** **(**extra **==** 1**)** **{**  m **-=** 1**;**  maxMulti **=** **(**int**)** Math**.**pow**(**3**,** m**);**  maxMulti **\*=** 4**;**  **}** **else** **{**  **if** **(**extra **==** 0**)** **{**  maxMulti **=** **(**int**)** Math**.**pow**(**3**,** m**);**  **}else** **{**  maxMulti **=** **(**int**)** Math**.**pow**(**3**,** m**)** **\*** extra**;**  **}**  **}**  **}else** **{**  int a **=** input **/** 2**;**  maxMulti **=** a **\*** **(**input **-**a**);**  **}**  **return** maxMulti**;**  **}**    @Test  public void testInput**()** **{**  System**.**out**.**println**(**maxMutli**()** **+** " " **+** expected**);**  assertEquals**(**maxMutli**(),** expected**);**  **}**    **}** |