1. 测试
2. 概述

我们写完程序后都会对程序的逻辑功能进行测试，看看程序是否有缺陷。通过测试，我们可以找出缺陷，完善程序的逻辑。

1. 测试分类

测试分为黑盒测试和白盒测试，黑盒测试只需要输入数据，得出输出值，而不关注具体的执行过程；白盒测试需要关注具体的执行过程，通过写代码来测试。

1. Junit
2. 概述

测试单元，当我们自己对程序进行测试时，都要重写一个测试类，且需通过main函数来测试。Junit单元允许我们只测试一个单元，如测试一个方法，测试一个类，这样测试就非常地轻便，简洁。

1. Junit使用方法
2. 定义测试类

建议将测试类都放在一个test包中，每个测试类的类名按照：被测试类+Test 的格式创建。

1. 定义测试方法

测试类中的方法，测试方法名按照 test+方法名 的格式创建，一般不需返回值，也不需要参数，因为只是用于测试方法的正确性。

1. 为测试方法添加注解@Test，并导入Junit依赖环境。
2. 在测试方法中添加测试代码

一般使用断言操作来判断结果，断言即判断正确结果与真实结果是否匹配：

Assert.assertEquals(期望，真实)

（三）Junit使用代码示例：

package test;  
  
import org.junit.Assert;  
import org.junit.Test;  
import 测试.Calculator;  
  
public class CalculatorTest {  
  
 @Test  
 public void testAdd(){  
 Calculator c = new Calculator();  
 int res = c.*add*(1, 2);  
 // System.out.println(res);  
  
 //断言  
 Assert.*assertEquals*(3, res);  
 }  
}

（四）注解

1. @Before

测试类中一般在测试方法执行之前都要申请资源，即在测试方法之前执行，可以用注解@Before来表示。

@Before  
public void init(){  
 System.*out*.println("init");  
}

2、@After

在测试之后，需要释放资源，用注解@After来表示在测试代码后执行的代码。

@After  
public void close(){  
 System.*out*.println("close");  
}

3. @BeforeClass

针对所有的测试，只执行一次，且必须为static void,优先级最高。

4.@AfterClass

针对所有的测试，只执行一次，且必须为static void,在最后执行。

1. [@Ignore](mailto:5.@Ignore)

忽略的测试方法

（五）单元测试用例的执行顺序

@BeforeClass –> @Before –> @Test –> @After –> @AfterClass