**《软件测试》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 姓名 | | 谢双骏 | | 年级 | | 2019级 |
| 学号 | | 20194196 | | 专业、班级 | | 计算机科学与技术01班 |
| 实验名称 | 实验一 单元测试 | | | | | |
| 实验时间 | 2022.4.12 | | 实验地点 | | DS3401 | |
| 实验成绩 |  | | 实验性质 | | □验证性 □设计性 □综合性 | |
| 教师评价：  评价教师签名： | | | | | | |
| 一、实验目的   1. 验证所学理论，巩固所学知识并加深理解； 2. 培养学生分析单元模块，设计测试数据的的能力； 3. 熟悉白盒测试、黑盒测试用例的设计； 4. 熟悉使用Junit框架进行基于java语言的单元测试 | | | | | | |
| 二、实验项目内容   1. 使用白盒测试用例设计方法为下面的程序设计测试用例： 2. 程序要求：10个铅球中有一个假球（比其他铅球的重量要轻），用天平三次称出假球。 3. 程序设计思路：第一次使用天平分别称5个球，判断轻的一边有假球；拿出轻的5个球，取出其中4个第二次称，两边分别放2个球：如果两边同重，则剩下的球为假球；若两边不同重，拿出轻的两个球称第三次，轻的为假球。 4. 使用等价类划分法设计下面的测试用例：   输入三个数作为边，分别满足一般三角形、等腰三角形和等边三角形。 | | | | | | |
| 1. 实验过程或算法（源程序） 2. 白盒测试 3. 程序代码   首先编写寻找假球算法的代码，算法思想非常简单，每次对半取，哪边轻则假球在哪边，一共取不超过三次。代码如下：   |  | | --- | | package unit\_test;  public class findball {      private static int weight[]=new int[10];      public findball(){}      public void set\_weight(int input\_weight[]){          for(int i=0;i<input\_weight.length;i++){               weight[i]=input\_weight[i];          }      }      //      public String find\_bogus\_ball(){          if(weight[0]+weight[1]+weight[2]+weight[3]+weight[4]<weight[5]+weight[6]+weight[7]+weight[8]+weight[9]){              if(weight[1]+weight[2]==weight[3]+weight[4]){                  return "1号球是假球！";              }              if(weight[1]+weight[2]<weight[3]+weight[4]){                  if (weight[1]<weight[2]) {                      return "2号球是假球！";                  }                  else {                      return "3号球是假球！";                  }              }              else {                  if (weight[3]<weight[4]){                      return "4号球是假球！";                  }                  else{                      return "5号球是假球！";                  }              }          }          else {              if(weight[6]+weight[7]==weight[8]+weight[9]){                  return "6号球是假球！";              }              if(weight[6]+weight[7]<weight[8]+weight[9]) {                  if (weight[6]<weight[7]) {                      return "7号球是假球！";                  }                  else {                      return "8号球是假球！";                  }              }              else {                  if (weight[8]<weight[9]) {                      return "9号球是假球！";                  }                  else {                      return "10号球是假球！";                  }              }          }       }  } |  1. 设计测试用例 2. 列出所有条件   A：weight[0]+weight[1]+weight[2]+weight[3]+weight[4]<  weight[5]+weight[6]+weight[7]+weight[8]+weight[9]  B：weight[1]+weight[2]==weight[3]+weight[4]  C：weight[1]+weight[2]<weight[3]+weight[4]  D：weight[1]<weight[2]  E：weight[3]<weight[4]  F：weight[6]+weight[7]==weight[8]+weight[9]  G：weight[6]+weight[7]<weight[8]+weight[9]  H：weight[6]<weight[7]  I：weight[8]<weight[9]   1. 画出流程图   wps   1. 流图   wps   1. 测试用例  |  |  |  | | --- | --- | --- | | **用例输入** | **期待输出** | **覆盖路径** | | {2，9，9，9，9，9，9，9，9，9} | 1号是假球 | 1-2-3 | | {9，2，9，9，9，9，9，9，9，9} | 2号是假球 | 1-2-4-5-7 | | {9，9，2，9，9，9，9，9，9，9} | 3号是假球 | 1-2-4-5-8 | | {9，9，9，2，9，9，9，9，9，9} | 4号是假球 | 1-2-4-6-9 | | {9，9，9，9，2，9，9，9，9，9} | 5号是假球 | 1-2-4-6-10 | | {9，9，9，9，9，2，9，9，9，9} | 6号是假球 | 1-11-12 | | {9，9，9，9，9，9，2，9，9，9} | 7号是假球 | 1-11-13-14-16 | | {9，9，9，9，9，9，9，2，9，9} | 8号是假球 | 1-11-13-14-17 | | {9，9，9，9，9，9，9，9，2，9} | 9号是假球 | 1-11-13-15-18 | | {9，9，9，9，9，9，9，9，9，2} | 10号是假球 | 1-11-13-15-19 |  1. 测试代码  |  | | --- | | package unit\_test;  import junit.framework.TestCase;  public class findball\_test extends TestCase{      int input\_weight1[]= {2,9,9,9,9,9,9,9,9,9};      int input\_weight2[]= {9,2,9,9,9,9,9,9,9,9};      int input\_weight3[]= {9,9,2,9,9,9,9,9,9,9};      int input\_weight4[]= {9,9,9,2,9,9,9,9,9,9};      int input\_weight5[]= {9,9,9,9,2,9,9,9,9,9};      int input\_weight6[]= {9,9,9,9,9,2,9,9,9,9};      int input\_weight7[]= {9,9,9,9,9,9,2,9,9,9};      int input\_weight8[]= {9,9,9,9,9,9,9,2,9,9};      int input\_weight9[]= {9,9,9,9,9,9,9,9,2,9};      int input\_weight10[]= {9,9,9,9,9,9,9,9,9,2};        public void test\_findball() {          findball fb= new findball();          fb.set\_weight(input\_weight1);          assertEquals("1号球是假球！",fb.find\_bogus\_ball());          fb.set\_weight(input\_weight2);          assertEquals("2号球是假球！",fb.find\_bogus\_ball());          fb.set\_weight(input\_weight3);          assertEquals("3号球是假球！",fb.find\_bogus\_ball());          fb.set\_weight(input\_weight4);          assertEquals("4号球是假球！",fb.find\_bogus\_ball());          fb.set\_weight(input\_weight5);          assertEquals("5号球是假球！",fb.find\_bogus\_ball());          fb.set\_weight(input\_weight6);          assertEquals("6号球是假球！",fb.find\_bogus\_ball());          fb.set\_weight(input\_weight7);          assertEquals("7号球是假球！",fb.find\_bogus\_ball());          fb.set\_weight(input\_weight8);          assertEquals("8号球是假球！",fb.find\_bogus\_ball());          fb.set\_weight(input\_weight9);          assertEquals("9号球是假球！",fb.find\_bogus\_ball());          fb.set\_weight(input\_weight10);          assertEquals("10号球是假球！",fb.find\_bogus\_ball());      }      public static void main(String args[]) {          junit.textui.TestRunner.run(findball\_test.class);      }  } |  1. 等价类划分设计测试用例 2. 程序代码  |  | | --- | | package unit\_test;  public class triangle\_judge {      public String judge\_triangle(int a,int b,int c){            if(a+b>c&&a+c>b&&b+c>a&&a>0&&b>0&&c>0){              if(a==b&&a==c){                  return "等边三角形";              }              else if(a==b||b==c||a==c){                  return "等腰三角形";              }              else{                  return "普通三角形";              }          }          else{              return "无法构成三角形";          }      }  } |  1. 设计测试用例 2. 划分等价类  |  |  |  | | --- | --- | --- | | 测试项 | 有效等价类 | 无效等价类 | | 三角形的三条边 | a>0(1) | a<=0(7) | | b>0(2) | b<=0(8) | | c>0(3) | c<=0(9) | | a+b>c(4) | a+b<=c(10) | | a+c>b(5) | a+c<=b(11) | | b+c>a(6) | b+c<=a(12) | | 等腰三角形 | a=b(13) | a!=b&&a!=c&&b!=c(16) | | a=c(14) | | b=c(15) | | 等边三角形 | a=b=c(17) | a!=b(18)  a!=c(19)  b!=c(20) |  1. 设计测试用例  |  |  |  |  | | --- | --- | --- | --- | | 序号 | a b c | 覆盖等价类 | 期望输出 | | 1 | 7, 8, 9 | 1、2、3、4、5、6、16、18、19、20 | 一般三角形 | | 2 | 0, 8, 9 | 7 | 不能组成三角形 | | 3 | 8, 0, 9 | 8 | 不能组成三角形 | | 4 | 8, 9, 0 | 9 | 不能组成三角形 | | 5 | 1, 8, 9 | 10 | 不能组成三角形 | | 6 | 8, 9, 1 | 11 | 不能组成三角形 | | 7 | 9, 8, 1 | 12 | 不能组成三角形 | | 8 | 8, 8, 9 | 13 | 等腰三角形 | | 9 | 8, 9, 8 | 14 | 等腰三角形 | | 10 | 9, 8, 8 | 15 | 等腰三角形 | | 11 | 9, 9, 9 | 17 | 等边三角形 |  1. 测试代码  |  | | --- | | package unit\_test;  import junit.framework.TestCase;  public class triangle\_judge\_test extends TestCase{      public void test\_triangle\_judge(){          triangle\_judge tj=new triangle\_judge();          assertEquals("普通三角形", tj.judge\_triangle(7, 8, 9));          assertEquals("无法构成三角形", tj.judge\_triangle(0, 8, 9));          assertEquals("无法构成三角形", tj.judge\_triangle(8, 0, 9));          assertEquals("无法构成三角形", tj.judge\_triangle(8, 9, 0));          assertEquals("无法构成三角形", tj.judge\_triangle(1, 8, 9));          assertEquals("无法构成三角形", tj.judge\_triangle(8, 9, 1));          assertEquals("无法构成三角形", tj.judge\_triangle(9, 8, 1));          assertEquals("等腰三角形", tj.judge\_triangle(8, 8, 9));          assertEquals("等腰三角形", tj.judge\_triangle(8, 9, 8));          assertEquals("等腰三角形", tj.judge\_triangle(9, 8, 8));          assertEquals("等边三角形", tj.judge\_triangle(9, 9, 9));    }      public static void main(String args[]) {          junit.textui.TestRunner.run(triangle\_judge\_test.class);      }  } | | | | | | | |
| 1. 实验结果及分析和（或）源程序调试过程 2. 白盒测试 3. 所有测试样例正确   球对   1. 更改第一个测试用例为第二个球轻，然后断言第一个球轻，执行结果错误   球错   1. 等价类划分 2. 所有测试样例正确   三角形对   1. 更改第一个测试用例为普通三角形三边，然后断言是等边三角形，执行结果错误   三角形错 | | | | | | |