designTest

Eachchoice

**@Test**

**public void ISP\_testTriang\_1(){**

**assertEquals(3,tri.Triang(1,1,1));**

**assertEquals(1,tri.Triang(4,3,2));**

**assertEquals(1,tri.Triang(4,2,3));**

**assertEquals(1,tri.Triang(3,4,2));**

**assertEquals(1,tri.Triang(3,2,4));**

**assertEquals(1,tri.Triang(4,2,3));**

**assertEquals(1,tri.Triang(4,3,2));**

**assertEquals(2,tri.Triang(2,2,3));**

**assertEquals(2,tri.Triang(2,3,2));**

**assertEquals(2,tri.Triang(3,2,2));**

**assertEquals(4,tri.Triang(2,3,0));**

**assertEquals(4,tri.Triang(0,3,2));**

**assertEquals(4,tri.Triang(2,0,3));**

**assertEquals(4,tri.Triang(2,0,0));**

**assertEquals(4,tri.Triang(0,2,0));**

**assertEquals(4,tri.Triang(0,0,2));**

**assertEquals(4,tri.Triang(0,0,0));**

**}**

Pair-wise

**@Test**

**public void ISP\_testTriang\_1(){**

**assertEquals(3,tri.Triang(1,1,1));**

**assertEquals(1,tri.Triang(2,3,4));**

**assertEquals(1,tri.Triang(3,4,2));**

**assertEquals(1,tri.Triang(4,2,3));**

**assertEquals(2,tri.Triang(2,2,3));**

**assertEquals(2,tri.Triang(2,3,2));**

**assertEquals(2,tri.Triang(3,2,2));**

**assertEquals(4,tri.Triang(2,3,0));**

**assertEquals(4,tri.Triang(0,3,2));**

**assertEquals(4,tri.Triang(2,0,3));**

**assertEquals(4,tri.Triang(2,0,0));**

**assertEquals(4,tri.Triang(0,2,0));**

**assertEquals(4,tri.Triang(0,0,2));**

**assertEquals(4,tri.Triang(0,0,0));**

**}**

Node

**@Test**

**public void ISP\_testTriang\_1(){**

**assertEquals(4,tri.Triang(0,0,1));**

**assertEquals(3,tri.Triang(2,2,2));**

**assertEquals(4,tri.Triang(2,2,4));**

**assertEquals(1,tri.Triang(2,4,5));**

**assertEquals(2,tri.Triang(2,2,3));**

**assertEquals(2,tri.Triang(2,3,2));**

**assertEquals(2,tri.Triang(3,2,2));**

**}**

**Edge**

**@Test**

**public void ISP\_testTriang\_1(){**

**assertEquals(4,tri.Triang(0,0,1));**

**assertEquals(3,tri.Triang(2,2,2));**

**assertEquals(4,tri.Triang(2,2,4));**

**assertEquals(1,tri.Triang(2,4,5));**

**assertEquals(2,tri.Triang(2,2,3));**

**assertEquals(2,tri.Triang(2,3,2));**

**assertEquals(2,tri.Triang(3,2,2));**

**}**

**Prime path**

**@Test**

**public void ISP\_testTriang\_1(){**

**assertEquals(4,tri.Triang(0,0,1));**

**assertEquals(3,tri.Triang(2,2,2));**

**assertEquals(4,tri.Triang(2,2,4));**

**assertEquals(1,tri.Triang(2,4,5));**

**assertEquals(2,tri.Triang(2,2,3));**

**assertEquals(2,tri.Triang(2,3,2));**

**assertEquals(2,tri.Triang(3,2,2));**

**}**

**All-use**

**@Test**

**public void ISP\_testTriang\_1(){**

**assertEquals(3,tri.Triang(2,2,2));**

**assertEquals(2,tri.Triang(2,2,3));**

**assertEquals(2,tri.Triang(2,3,2));**

**assertEquals(2,tri.Triang(3,2,2));**

**}**

Logic coverage(Triang)

**public void ISP\_testTriang\_1(){**

**assertEquals(4,tri.Triang(-1,-2,-3));**

**assertEquals(4,tri.Triang(-1,2,3));**

**assertEquals(4,tri.Triang(-1,2,-3));**

**assertEquals(4,tri.Triang(1,2,-3));**

**assertEquals(4,tri.Triang(1,-2,-3));**

**assertEquals(4,tri.Triang(0,-3,2));**

**assertEquals(4,tri.Triang(1,1,2));**

**assertEquals(4,tri.Triang(1,-2,3));**

**assertEquals(1,tri.Triang(2,3,4));**

**assertEquals(1,tri.Triang(2,4,3));**

**assertEquals(1,tri.Triang(3,2,4));**

**assertEquals(1,tri.Triang(3,4,2));**

**assertEquals(1,tri.Triang(4,2,3));**

**assertEquals(1,tri.Triang(4,3,2));**

**assertEquals(4,tri.Triang(5,3,2));**

**assertEquals(3,tri.Triang(3,3,3));**

**assertEquals(4,tri.Triang(5,1,1));**

**assertEquals(4,tri.Triang(1,5,1));**

**assertEquals(4,tri.Triang(1,1,5));**

**}**

TritypMutantOne.java

**public void ISP\_testTriang\_1(){**

**assertEquals(4,tri.Triang(-1,-2,-3));**

**assertEquals(4,tri.Triang(-1,2,3));**

**assertEquals(4,tri.Triang(-1,2,-3));**

**assertEquals(4,tri.Triang(1,2,-3));**

**assertEquals(4,tri.Triang(1,-2,-3));**

**assertEquals(4,tri.Triang(0,-3,2));**

**assertEquals(4,tri.Triang(1,1,2));//错误发生**

**assertEquals(4,tri.Triang(1,-2,3));**

**assertEquals(1,tri.Triang(2,3,4));**

**assertEquals(1,tri.Triang(2,4,3));**

**assertEquals(1,tri.Triang(3,2,4));**

**assertEquals(1,tri.Triang(3,4,2));**

**assertEquals(1,tri.Triang(4,2,3));**

**assertEquals(1,tri.Triang(4,3,2));**

**assertEquals(4,tri.Triang(5,3,2));**

**assertEquals(3,tri.Triang(3,3,3));**

**assertEquals(4,tri.Triang(5,1,1));**

**assertEquals(4,tri.Triang(1,5,1));**

**assertEquals(4,tri.Triang(1,1,5));**

**}**

TritypMutantTwo.java

**public void ISP\_testTriang\_1(){**

**assertEquals(4,tri.Triang(-1,-2,-3));**

**assertEquals(4,tri.Triang(-1,2,3));**

**assertEquals(4,tri.Triang(-1,2,-3));**

**assertEquals(4,tri.Triang(1,2,-3));**

**assertEquals(4,tri.Triang(1,-2,-3));**

**assertEquals(4,tri.Triang(0,-3,2));**

**assertEquals(4,tri.Triang(1,1,2));//错误发生**

**assertEquals(4,tri.Triang(1,-2,3));**

**assertEquals(1,tri.Triang(2,3,4));**

**assertEquals(1,tri.Triang(2,4,3));**

**assertEquals(1,tri.Triang(3,2,4));**

**assertEquals(1,tri.Triang(3,4,2));**

**assertEquals(1,tri.Triang(4,2,3));**

**assertEquals(1,tri.Triang(4,3,2));**

**assertEquals(4,tri.Triang(5,3,2));**

**assertEquals(3,tri.Triang(3,3,3));**

**assertEquals(4,tri.Triang(5,1,1));**

**assertEquals(4,tri.Triang(1,5,1));**

**assertEquals(4,tri.Triang(1,1,5));**

**}**