**Module 2**

**2(a) BVA for the Triangle Problem**

1. Triangle\_SN

import static org.junit.Assert.\*;

import org.junit.Test;

public class Triangle\_SN

{

@Test

public void test5()

{

Triangle t=new Triangle();

assertEquals(t.check(3,3,3),"Equilateral");

}

@Test

public void test6()

{

Triangle t=new Triangle();

assertEquals(t.check(2,2,3),"Isoceles");

}

@Test

public void test7()

{

Triangle t=new Triangle();

assertEquals(t.check(3,4,5),"Scalene");

}

@Test

public void test8()

{

Triangle t=new Triangle();

assertEquals(t.check(4,1,2),"Not a Triangle");

}

}

1. Triangle\_SR

import static org.junit.Assert.\*;

import org.junit.Test;

public class Triangle\_SR

{

@Test

public void test13()

{

Triangle t=new Triangle();

assertEquals(t.check(-1,-1,5),"Invalid Values");

}

@Test

public void test14()

{

Triangle t=new Triangle();

assertEquals(t.check(5,-1,-1),"Invalid Values");

}

@Test

public void test15()

{

Triangle t=new Triangle();

assertEquals(t.check(-1,5,-1),"Invalid Values");

}

@Test

public void test16()

{

Triangle t=new Triangle();

assertEquals(t.check(-1,-1,-1),"Invalid Values");

}

@Test

public void test17()

{

Triangle t=new Triangle();

assertEquals(t.check(-1,5,5),"Invalid Values");

}

@Test

public void test18()

{

Triangle t=new Triangle();

assertEquals(t.check(5,-1,5),"Invalid Values");

}

@Test

public void test19()

{

Triangle t=new Triangle();

assertEquals(t.check(5,5,-1),"Invalid Values");

}

@Test

public void test20()

{

Triangle t=new Triangle();

assertEquals(t.check(5,5,5),"Equilateral");

}

}

1. Triangle\_WN

import static org.junit.Assert.\*;

import org.junit.Test;

public class Triangle\_WN

{

@Test

public void test1()

{

Triangle t=new Triangle();

assertEquals(t.check(5,5,5),"Equilateral");

}

@Test

public void test2()

{

Triangle t=new Triangle();

assertEquals(t.check(2,2,3),"Isoceles");

}

@Test

public void test3()

{

Triangle t=new Triangle();

assertEquals(t.check(3,4,5),"Scalene");

}

@Test

public void test4()

{

Triangle t=new Triangle();

assertEquals(t.check(4,1,2),"Not a Triangle");

}

}

1. Triangle\_WR

import static org.junit.Assert.\*;

import org.junit.Test;

public class Triangle\_WR

{

@Test

public void test9()

{

Triangle t=new Triangle();

assertEquals(t.check(201,5,5),"Invalid Values");

}

@Test

public void test10()

{

Triangle t=new Triangle();

assertEquals(t.check(5,201,5),"Invalid Values");

}

@Test

public void test11()

{

Triangle t=new Triangle();

assertEquals(t.check(5,5,201),"Invalid Values");

}

@Test

public void test12()

{

Triangle t=new Triangle();

assertEquals(t.check(2,2,3),"Isoceles");

}

}

**2(b)Equivalence Class Partitioning For The Triangle Problem**

import static org.junit.Assert.\*;

import org.junit.Test;

public class TriangleTest

{

@Test

public void test2a\_1()

{

Triangle t=new Triangle();

assertEquals(t.check(1,100,100),"Isoceles");

}

@Test

public void test2a\_2()

{

Triangle t=new Triangle();

assertEquals(t.check(2,100,100),"Isoceles");

}

@Test

public void test2a\_3()

{

Triangle t=new Triangle();

assertEquals(t.check(100,100,100),"Equilateral");

}

@Test

public void test2a\_4()

{

Triangle t=new Triangle();

assertEquals(t.check(199,100,100),"Isoceles");

}

@Test

public void test2a\_5()

{

Triangle t=new Triangle();

assertEquals(t.check(200,100,100),"Not a Triangle");

}

@Test

public void test2a\_6()

{

Triangle t=new Triangle();

assertEquals(t.check(100,1,100),"Isoceles");

}

@Test

public void test2a\_7()

{

Triangle t=new Triangle();

assertEquals(t.check(100,2,100),"Isoceles");

}

@Test

public void test2a\_8()

{

Triangle t=new Triangle();

assertEquals(t.check(100,100,100),"Equilateral");

}

@Test

public void test2a\_9()

{

Triangle t=new Triangle();

assertEquals(t.check(100,199,100),"Isoceles");

}

@Test

public void test2a\_10()

{

Triangle t=new Triangle();

assertEquals(t.check(100,200,100),"Not a Triangle");

}

@Test

public void test2a\_11()

{

Triangle t=new Triangle();

assertEquals(t.check(100,100,1),"Isoceles");

}

@Test

public void test2a\_12()

{

Triangle t=new Triangle();

assertEquals(t.check(100,100,2),"Isoceles");

}

@Test

public void test2a\_13()

{

Triangle t=new Triangle();

assertEquals(t.check(100,100,100),"Equilateral");

}

@Test

public void test2a\_14()

{

Triangle t=new Triangle();

assertEquals(t.check(100,100,199),"Isoceles");

}

@Test

public void test2a\_15()

{

Triangle t=new Triangle();

assertEquals(t.check(100,100,200),"Not a Triangle");

}

}

**2(c) BVA For NextDate Function**

***NextDate Program***

**package** module3;

**public** **class** NextDate

{

**public** **static** String next(**int** d,**int** m,**int** y,**int** cc)

{

**if**(d==cc)

{

d=1;

**if**(m==12)

{

y++;

m=1;

}

**else**

{

m++;

}

}

**else** **if**(d<cc)

{

d++;

}

**else**

{

**return** "Invalid Values";

}

**return** (String.*valueOf*(d)+"."+String.*valueOf*(m)+"."+String.*valueOf*(y));

}

**public** String nextday(**int** d,**int** m,**int** y)

{

**if**(d>=1 && d<=31 && m>=1 && m<=12 && y>=1819 && y<=2019)

{

**switch**(m)

{

**case** 1 :

**case** 3 :

**case** 5 :

**case** 7 :

**case** 8 :

**case** 10:

**case** 12: **return**(*next*(d,m,y,31));

**case** 4 :

**case** 6 :

**case** 9 :

**case** 11: **return**(*next*(d,m,y,30));

**default** : **return**(*next*(d,m,y,((y%4==0 && y%100!=0) || y%400==0)?29:28));

}

}

**return** "Invalid Values";

}

}

1. NextDate\_SN

import static org.junit.Assert.\*;

import org.junit.Test;

public class NextDate\_SN

{

@Test

public void test2d\_1()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(1,6,1919),"2.6.1919");

}

@Test

public void test2d\_2()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,1,1819),"17.1.1819");

}

@Test

public void test2d\_3()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,1819),"17.6.1819");

}

@Test

public void test2d\_4()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,2009),"17.6.2009");

}

}

1. NextDate\_SR

import static org.junit.Assert.\*;

import org.junit.Test;

public class NextDate\_SR

{

@Test

public void test2d\_13()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(-1,6,1919),"Invalid Input");

}

@Test

public void test2d\_14()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,-1,1819),"Invalid Input");

}

@Test

public void test2d\_15()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(-1,-1,1919),"Invalid Input");

}

@Test

public void test2d\_16()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(32,-1,2009),"Invalid Input");

}

@Test

public void test2d\_17()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,-1,2020),"Invalid Input");

}

@Test

public void test2d\_18()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(-1,6,2020),"Invalid Input");

}

@Test

public void test2d\_19()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(-1,-1,2020),"Invalid Input");

}

@Test

public void test2d\_20()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,2009),"17.6.2009");

}

}

1. NextDate\_WN

import static org.junit.Assert.\*;

import org.junit.Test;

public class NextDate\_WN

{

@Test

public void test2d\_1()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(1,6,1919),"2.6.1919");

}

@Test

public void test2d\_2()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,1,1819),"17.1.1819");

}

@Test

public void test2d\_3()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,1819),"17.6.1819");

}

@Test

public void test2d\_4()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,2009),"17.6.2009");

}

}

1. NextDate\_WR

import static org.junit.Assert.\*;

import org.junit.Test;

public class NextDate\_WR

{

@Test

public void test2d\_9()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(0,6,1919),"Invalid Input");

}

@Test

public void test2d\_10()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,0,1819),"Invalid Input");

}

@Test

public void test2d\_11()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,2020),"Invalid Input");

}

@Test

public void test2d\_12()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,2009),"17.6.2009");

}

}

**2(d)Equivalence Class Partitioning For The NextDate Function**

import static org.junit.Assert.\*;

import org.junit.Test;

public class NextDateTest

{

@Test

public void test2b\_1()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(1,6,1919),"2.6.1919");

}

@Test

public void test2b\_2()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,1,1819),"17.1.1819");

}

@Test

public void test2b\_3()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,1819),"17.6.1819");

}

@Test

public void test2b\_4()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,2009),"17.6.2009");

}

@Test

public void test2b\_5()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(1,6,1919),"2.6.1919");

}

@Test

public void test2b\_6()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,1,1819),"17.1.1819");

}

@Test

public void test2b\_7()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,1819),"17.6.1819");

}

@Test

public void test2b\_8()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,2009),"17.6.2009");

}

@Test

public void test2b\_9()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(0,6,1919),"Invalid Input");

}

@Test

public void test2b\_10()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,0,1819),"Invalid Input");

}

@Test

public void test2b\_11()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,2020),"Invalid Input");

}

@Test

public void test2b\_12()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,2009),"17.6.2009");

}

@Test

public void test2b\_13()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(-1,6,1919),"Invalid Input");

}

@Test

public void test2b\_14()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,-1,1819),"Invalid Input");

}

@Test

public void test2b\_15()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(-1,-1,1919),"Invalid Input");

}

@Test

public void test2b\_16()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(32,-1,2009),"Invalid Input");

}

@Test

public void test2b\_17()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,-1,2020),"Invalid Input");

}

@Test

public void test2b\_18()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(-1,6,2020),"Invalid Input");

}

@Test

public void test2b\_19()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(-1,-1,2020),"Invalid Input");

}

@Test

public void test2b\_20()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,6,2009),"17.6.2009");

}

@Test

public void test2b\_21()

{

NextDate obj= new NextDate();

assertEquals(obj.nextdate(16,3,2018),"17.3.2018");

}

}