Junit Assignment

1.

* MinMaxFinder.java

import java.util.Arrays;

import java.util.List;

import java.util.Scanner;

class MinMaxFinder{

public int[] find()

{

int[] numbers = new int[9];

Scanner scan = new Scanner(System.in);

System.out.println("Input 9 numbers");

for(int i=0;i<9;i++)

{

numbers[i] = scan.nextInt();

}

int min = numbers[0];

int max = numbers[0];

for(int i=0;i<9;i++)

{

if(min>numbers[i])

{

min = numbers[i];

}

if(max<numbers[i])

{

max = numbers[i];

}

}

// System.out.println("{"+min+","+max+"}");

int[] arr= {min,max};

return arr;

}}

* MinMaxFinderTest.java

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.Assertions;

import org.junit.jupiter.api.Test;

class MinMaxFinderTest {

@Test

void testforEquality() {

MinMaxFinder minmax=new MinMaxFinder();

int[] expected=new int[]{23,56};

int[] actual=minmax.find();

Assertions.assertArrayEquals(expected,actual);

}

@Test

void testforNotEqual() {

MinMaxFinder minmax=new MinMaxFinder();

int[] expected=new int[]{23,56};

int[] actual=minmax.find();

Assertions.assertNotEquals(expected,actual);

}

@Test

void test() {

MinMaxFinder minmax=new MinMaxFinder();

boolean res=true;

int[] expected=new int[]{23,56};

int[] actual=minmax.find();

if(actual != null) {

res=false;

}

Assertions.assertFalse(res);

}

}

2.

3.

* BankMain.java

**public** **class** BankMain {

String name;

//int acc\_id;

**int** balance;

**int** withdraw;

**public** BankMain(String name, **int** balance, **int** withdraw) {

**super**();

**this**.name = name;

//this.acc\_id = acc\_id;

**this**.balance = balance;

**this**.withdraw = withdraw;

}

**public** **int** getBalance() **throws** InsufficientBalanceException {

**if**(balance>withdraw) {

balance=balance-withdraw;

System.***out***.println(balance);

**return** balance;

}

**else** {

**throw** **new** InsufficientBalanceException("Insufficient Balance"+balance);

}

}

}

* InsufficientBalanceException.java

**public** **class** InsufficientBalanceException **extends** Exception{

**public** InsufficientBalanceException(String message) {

**super**(message);

}

}

* BankMainTest.java

import static org.junit.jupiter.api.Assertions.assertThrows;

import static org.junit.Assert.assertThrows;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.Test;

import org.junit.function.ThrowingRunnable;

import org.junit.jupiter.api.Assertions;

import org.junit.jupiter.api.function.Executable;

class BankMainTest {

@Test

void test() throws InsufficientBalanceException {

BankMain bankmain=new BankMain("Pragati", 1000, 90);

Assertions.assertThrows(InsufficientBalanceException.class,()->bankmain.getBalance());

System.out.println("Balance:"+bankmain.getBalance());

//Throwable exception = assertThrows(InsufficientBalanceException.class,

// (ThrowingRunnable) ()->{bankmain.getBalance();});

}

}

4.

* Basic.java

public class Basic {

public int add(int a,int b) {

return a+b;

}

public int substract(int p,int q) {

return p-q;

}

}

* BasicTest.java

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.AfterAll;

import org.junit.jupiter.api.AfterEach;

import org.junit.jupiter.api.Assertions;

import org.junit.jupiter.api.BeforeAll;

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.Test;

class BasicTest {

Basic basic;

static @BeforeAll

void Befinit() {

System.out.println("Started");

}

static @AfterAll

void Afinit() {

System.out.println("Completed");

}

@BeforeEach

void init() {

basic = new Basic();

}

@AfterEach

void initafter() {

System.out.println("Done");

}

@Test

void testadd() {

//Basic basic=new Basic();

int expected=3;

int actual=basic.add(1,2);

Assertions.assertEquals(expected, actual);

}

@Test

void testsubs() {

//Basic basic=new Basic();

int expected=2;

int actual=basic.substract(3,1);

Assertions.assertEquals(expected, actual);

}

}