Q1

package cg;

public class MinMaxFinder {

int[] findMinMax(int[] arr){

int max=arr[0];

int min=arr[0];

for (int i:arr) {

if(i>max)

max=i;

if(i<min)

min=i;

}

return new int[]{min, max};

}

}

+++++++++++++++++++++++++++++++++++++++++++++

package cg;

import org.junit.jupiter.api.Test;

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

class MinMaxFinderTest {

MinMaxFinder a;

@Test

void findMinMax() {

a=new MinMaxFinder();

assertArrayEquals(new int[]{2,99}, a.findMinMax(new int[]{89,2,56,43,99}));

assertArrayEquals(new int[]{-1,9}, a.findMinMax(new int[]{5,6,0,7,-1,8,9}));

assertArrayEquals(new int[]{0,15}, a.findMinMax(new int[]{15, 8, 5, 3, 0}));

assertArrayEquals(new int[]{42,69}, a.findMinMax(new int[]{55,65,54,59,62,42,69}));

}

}

=================================================================================================

Q2

package cg;

public class MinMaxFinderObject {

String findMinMax(int[] arr){

int max=arr[0];

int min=arr[0];

for (int i:arr) {

if(i>max)

max=i;

if(i<min)

min=i;

}

return min+" "+max;

}

}

+++++++++++++++++++++++++++++++++++++++++=

package cg;

import org.junit.jupiter.api.Test;

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

class MinMaxFinderObjectTest {

MinMaxFinderObject o;

@Test

void findMinMax() {

o=new MinMaxFinderObject();

assertAll(

()->assertEquals("2 99",o.findMinMax(new int[]{89,2,56,43,99})),

()->assertEquals("-1 9",o.findMinMax(new int[]{5,6,0,7,-1,8,9})),

()->assertEquals("0 15",o.findMinMax(new int[]{15, 8, 5, 3, 0})),

()->assertEquals("42 69",o.findMinMax(new int[]{55,65,54,59,62,42,69})));

}

}

================================================================================================

Q3

package cg;

public class junit53 {

int balance=21000;

void withdraw(int m) throws InsufficientFundsException{

if(m>balance)

throw new InsufficientFundsException("Insufficient Balance");

}

}

class InsufficientFundsException extends Exception{

InsufficientFundsException(String str){

super(str);

}

}

+++++++++++++++++++++++++++++++++++++++++++=

package cg;

import org.junit.jupiter.api.Test;

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

class BankAccountTest {

junit53 b;

@Test

void withdraw() {

b=new junit53();

//balance is 21000

assertThrows(InsufficientFundsException.class,()->b.withdraw(30000));

assertThrows(InsufficientFundsException.class,()->b.withdraw(40000));

assertThrows(InsufficientFundsException.class,()->b.withdraw(25000));

// This Test fails since InsufficientFundsException is not thrown

// assertThrows(InsufficientFundsException.class,()->b.withdraw(100));

}

}

===================================================================================

Q4

package cg;

import java.util.stream.DoubleStream;

class Calc {

static double add(double... operands) {

return DoubleStream.of(operands).sum();

}

}

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

package cg;

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

class CalcTest {

@BeforeAll

static void init(){

System.out.println("@BeforeAll executed");

}

@BeforeEach

void startup(){

System.out.println("@BeforeEach executed");

}

@Test

@DisplayName("@Test1")

void testCalcOne()

{

System.out.println("======TEST ONE EXECUTED=======");

Assertions.assertEquals( 4 , Calc.add(2, 2));

}

@Test

@DisplayName("@Test2")

void testCalcTwo()

{

System.out.println("======TEST TWO EXECUTED=======");

Assertions.assertEquals( 6 , Calc.add(2, 4));

}

@AfterEach

void cleanup(){

System.out.println("@AfterEach executed");

}

@AfterAll

static void exit(){

System.out.println("@AfterAll executed");

}

}

===============================================================