Q1)

import java.util.ArrayList;

import java.util.Arrays;

public class MinMaxFinder

{

static int max;

static int min;

public static int[] find\_Min\_Max(int my\_array[])

{

max = my\_array[0];

min = my\_array[0];

int len = my\_array.length;

for (int i = 1; i < len - 1; i = i + 2)

{

if (i + 1 > len)

{

if (my\_array[i] > max) max = my\_array[i];

if (my\_array[i] < min) min = my\_array[i];

}

if (my\_array[i] > my\_array[i + 1])

{

if (my\_array[i] > max) max = my\_array[i];

if (my\_array[i + 1] < min) min = my\_array[i + 1];

}

if (my\_array[i] < my\_array[i + 1])

{

if (my\_array[i] < min) min = my\_array[i];

if (my\_array[i + 1] > max) max = my\_array[i + 1];

}

}

return my\_array;

}

public static void main(String[] args)

{

int[] my\_array = {25, 14, 56, 15, 36, 56, 77, 18, 29, 49};

find\_Min\_Max(my\_array);

System.out.println(" Original Array: "+Arrays.toString(my\_array));

ArrayList<Integer> list = new ArrayList<Integer>();

list.add(min);

list.add(max);

System.out.println("Minimum and Maximum values : " +list);

}

Q2)

import static org.junit.Assert.assertArrayEquals;

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

import org.junit.Assert;

import org.junit.Test;

import org.junit.jupiter.api.Assertions;

public class MinMaxFinderTest

{

@Test

public void testfind\_Min\_Max()

{

MinMaxFinder arr = new MinMaxFinder();

int[] expected= new int[] {2,20};

int[] expected1= new int[] {3,77};

int[] expected2= new int[] {4,50};

assertArrayEquals(new int[] {2,20},arr.find\_Min\_Max(new int[] {2,5,20}));

assertArrayEquals(new int[] {2,20},arr.find\_Min\_Max(new int[] {3,4,77}));

assertArrayEquals(new int[] {2,20},arr.find\_Min\_Max(new int[] {4,5,50}));

}

private void assertArrayEquals(int[] is, Object find\_Min\_Max)

{

System.out.println("AsserArrayEqual");

}

}

Q3)

import java.util.ArrayList;

import java.util.Arrays;

public class MinMax

{

static int max;

static int min;

public static int[] MinMax(int my\_array[])

{

max = my\_array[0];

min = my\_array[0];

int len = my\_array.length;

for (int i = 1; i < len - 1; i = i + 2)

{

if (i + 1 > len)

{

if (my\_array[i] > max) max = my\_array[i];

if (my\_array[i] < min) min = my\_array[i];

}

if (my\_array[i] > my\_array[i + 1])

{

if (my\_array[i] > max) max = my\_array[i];

if (my\_array[i + 1] < min) min = my\_array[i + 1];

}

if (my\_arra…

Q4)

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

import org.junit.jupiter.api.Test;

public class MinMaxTest

{

@Test

public void MinMaxtest()

{

MinMax arr = new MinMax();

int[] expected= new int[] {2,20};

int[] expected1= new int[] {1,15};

int[] expected2= new int[] {3,60};

assertArrayEquals(new int[] {2,20},arr.MinMax(new int[] {2,5,20}));

assertArrayEquals(new int[] {2,20},arr.MinMax(new int[] {1,2,15}));

assertArrayEquals(new int[] {2,20},arr.MinMax(new int[] {3,5,60}));

}

private void assertArrayEquals(int[] is, Object find\_Min\_Max)

{

System.out.println("AsserArrayEqual");

}

}

Q5)

import static org.junit.Assert.assertEquals;

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

import java.text.Collator;

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;

public class MathUtils

{

MathUtils mathUtils;

@BeforeAll

static void setup()

{

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

}

@BeforeEach

void init()

{

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

}

@Test

void testAdd()

{

System.out.println("Addition of two numbers");

}

public int add(int i, int j)

{

return i+j;

}

@Test

void testSub()

{

System.out.println("Subtraction of two numbers");

}

public int sub(int i, int j)

{

return i-j;

}

@AfterEach()

void tearThis()

{

System.out.println("AfterEach Executed");

}

@AfterAll()

static void tear()

{

System.out.println("AfterAll Executed");

}

}

Q6)

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

import org.junit.jupiter.api.Test;

public class MathutilsTest

{

@Test

void testAdd()

{

MathUtils mathUtils = new MathUtils();

int expected = 2;

int actual = mathUtils.add(1, 1);

assertEquals(expected,actual);

}

@Test

void testSub()

{

MathUtils mathUtils = new MathUtils();

int expected = 4;

int actual = mathUtils.sub(6, 2);

assertEquals(expected,actual);

}

}

Q7)

import javax.naming.InsufficientResourcesException;

public class Account

{

static int balance = 3000;

public static void main(String[] args) throws InsufficientResourcesException

{

Account ac = new Account();

int amount;

int withdraw = ac.withdraw(amount=2000);

System.out.println(balance);

}

public int withdraw(int amount) throws InsufficientResourcesException

{

if(balance<amount)

{

throw new InsufficientResourcesException(String.format(

"Current balance %d is less than requested amount %d",

balance,amount));

}

balance = balance-amount;

return balance;

}

}

Q8)

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

import javax.naming.InsufficientResourcesException;

import org.junit.jupiter.api.Test;

public class AccountTest

{

@Test

void array() throws InsufficientResourcesException

{

Account arr = new Account();

assertEquals(2000, arr.withdraw(1000));

}

}