Quetion1

**package** junitAssignment;

**import** java.util.Arrays;

**public** **class** MinMaxFinder {

**public** **int**[] arr(**int** [] numbers) {

Arrays.*sort*(numbers);

**int** [] arr1= {numbers[0],numbers[numbers.length-1]};

**return** arr1;

}

}

MinMaxTest class:

package junitAssignment;

import static org.junit.Assert.assertArrayEquals;

import org.junit.jupiter.api.Test;

public class MinMaxTest {

@Test

void testMinMaxFind() {

MinMaxFinder mnf = new MinMaxFinder();

int expedted[] = new int[] {3,56};

assertArrayEquals(expedted, mnf.arr(new int[] {56,34,7,3,54,3,34,34,53}));

}

@Test

void testMinMaxFind1() {

MinMaxFinder mnf1 = new MinMaxFinder();

int expedted1[] = new int[] {0,99};

assertArrayEquals(expedted1, mnf1.arr(new int[] {30,1,10,25,56,99,87,45,0}));

}

@Test

void testMinMaxFind2() {

MinMaxFinder mnf2 = new MinMaxFinder();

int expedted2[] = new int[] {101,999};

assertArrayEquals(expedted2, mnf2.arr(new int[] {999,101,205,665,777,854,465}));

}

}

Question2:

MinMax class:

**package** junitAssignment;

**public** **class** MinMax {

**public** **int** min,max;

MinMax(){}

MinMax(**int** min,**int** max){

**super**();

**this**.min=min;

**this**.max=max;

}

}

MinMaxObject class:

**package** junitAssignment;

**public** **class** MinMaxObject {

**public** MinMax findMinMax(**int** arr[]) {

MinMax obj=**new** MinMax();

obj.max=arr[1];

obj.min=arr[1];

**for**(**int** i=0;i<arr.length;i++) {

**if**(arr[i]<obj.min) {

obj.min=arr[i];

}

**if**(arr[i]>obj.max) {

obj.max=arr[i];

}

}

**return** obj;

}

}

MinMaxObjectTest class:

**package** junitAssignment;

**import** **static** org.junit.Assert.*assertEquals*;

**import** org.junit.Test;

**public** **class** MinMaxObjectTest {

**int** arr[]= {56,34,7,3,54,3,34,34,53};

@Test

**void** testFindMax() {

MinMax expectedObj=**new** MinMax(3,56);

MinMax actualObj;

MinMaxObject obj = **new** MinMaxObject();

actualObj=obj.findMinMax(arr);

*assertEquals*(actualObj.max,expectedObj.max);

*assertEquals*(actualObj.min,expectedObj.min);

}

}

Question3

BankAccount class:

**package** junitAssignment;

**import** javax.naming.InsufficientResourcesException;

**public** **class** BankAccount {

**int** a;

**int** BankAccountBalance = 20000;

**public** String Withdraw(**int** a) **throws** InsufficientResourcesException {

**if**(a< BankAccountBalance) {

**return** ("wait for a momment");

}

**else**

{

**throw** **new** InsufficientResourcesException("Insufficient Funds");

}

}

}

BankAccountTest class:

**package** junitAssignment;

**import** **static** org.junit.Assert.*assertEquals*;

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

**import** javax.naming.InsufficientResourcesException;

**import** org.junit.Test;

**public** **class** BankAccountTest {

@Test

**public** **void** testwithdraw() {

BankAccount a = **new** BankAccount();

*assertThrows*(InsufficientResourcesException.**class**, ()-> a.Withdraw(20000),"An Exception may be occurred" );

}

@Test

**public** **void** testwithdraw1() **throws** InsufficientResourcesException {

BankAccount a1 = **new** BankAccount();

String expected = "wait for a momment";

*assertEquals*(expected, a1.Withdraw(19999));

}

}

Question4

Example class:

**package** junitAssignment;

**public** **class** Example {

**public** **int** add (**int** a, **int** b) {

**return** a+b;

}

**public** **int** subtraction (**int** a, **int** b) {

**return** a-b;

}

}

ExampleTest class:

**package** junitAssignment;

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

**import** org.junit.Test;

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

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

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

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

**public** **class** ExampleTest {

Example junit;

@BeforeAll

**static** **void** beforeAllInit() {

System.***out***.println("this nedds to run before all");

}

@AfterAll

**static** **void** afterAll() {

System.***out***.println("We are at the end of the Programming");

}

@BeforeEach

**void** init() {

junit = **new** Example();

}

@AfterEach

**void** afterEach() {

System.***out***.println("The code run successfully");

}

@Test

**public** **void** addtest() {

**int** result = junit.add(10, 20);

*assertEquals*(30, result);

}

@Test

**public** **void** subtracttest() {

**int** result = junit.subtraction(10, 9);

*assertEquals*(1, result);

}

}