**Junit Assignment**

1. **package** NewPackageTest;

**public** **class** MinMaxFinder

{

**int**[] NewArr()

{

**int**[] element;

**int** [] arr = {1,23,45,7,456};

**int** max = arr[0];

**int** min = arr[0];

**for** (**int** i = 1; i < arr.length; i++)

{

// if the current element is greater than the maximum found so far

**if** (arr[i] > max)

{

max = arr[i];

}

// if the current element is smaller than the minimum found so far

**else** **if** (arr[i] < min)

{

min = arr[i];

}

}

**int** [] arr2 = {min, max};

element = arr2;

**return** element;

}

}

**MinMaxTestFinder:**

**package** NewPackageTest;

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

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

**class** MinMaxFinderTest {

@Test

**void** test()

{

MinMaxFinder mn = **new** MinMaxFinder();

**int** [] arr = {1,456};

**int**[] expected = arr;

**int**[] actual = mn.NewArr();

*assertArrayEquals*(expected, actual, "TestCase 1");

}

@Test

**void** test1()

{

MinMaxFinder mn = **new** MinMaxFinder();

**int** [] arr = {1,456,23};

**int**[] expected = arr;

**int**[] actual = mn.NewArr();

*assertArrayEquals*(expected, actual, "TestCase 2");

}

@Test

**void** test2()

{

MinMaxFinder mn = **new** MinMaxFinder();

**int** [] arr = {1,456,23,45};

**int**[] expected = arr;

**int**[] actual = mn.NewArr();

*assertArrayEquals*(expected, actual, "TestCase 3");

}

}

1. **package** NewPackageTest;

**import** java.util.Scanner;

**public** **class** BankAcount

{

**private** **int** balance;

**public** **void** BankOperations()

{

balance = 100;

}

**public** **int** getbalance()

{

**return** balance;

}

**public** **void** withdraw(**int** amount) **throws** InsufficientFundsException

{

**if** (amount < balance)

{

**int** newBalance = balance - amount;

}

**else**

{

**throw** **new** InsufficientFundsException(amount);

}

}

}

package NewPackageTest;

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

import org.junit.jupiter.api.Test;

class BankAcountTest {

@Test

public void setUp() throws InsufficientFundsException

{

BankAcount bn = new BankAcount();

assertThrows(InsufficientFundsException.class, () -> bn.withdraw(2));

}

}

**package** NewPackageTest;

**public** **class** InsufficientFundsException **extends** Exception

{

**private** **final** **int** amount;

**public** InsufficientFundsException(**int** amount)

{

**this**.amount = amount;

}

**public** **int** getAmt()

{

**return** amount;

}

}

1. **package** NewPackageTest;

**public** **class** ClassA

{

**void** abc()

{

System.***out***.println("This is the add method of Class A");

}

}

**package** NewPackageTest;

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

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

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

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

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

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

**class** ClassATest

{

ClassA abc;

@BeforeAll //#1

**static** **void** BeforeAll()

{

System.***out***.println("This Method will run Before all the methods present in the class");

}

@Test

**void** Test() //#3

{

System.***out***.println("Test");

}

@BeforeEach

**void** Method2() //#2

{

abc = **new** ClassA();

System.***out***.println("This is Method2 (Before Each)");

}

@AfterEach //#4

**void** Method3()

{

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

}

@Test //#5

**void** Test2()

{

System.***out***.println("Test2");

}

@Test //#5

**void** Test3()

{

System.***out***.println("Test3");

}

@AfterAll //#6

**static** **void** Method4()

{

System.***out***.println("This is Method4 (After All)");

}

}