JAVA TEST – 2

QUESTION – 1

```
package javatest2;
import java.util.*;
interface AdvancedArithmetic{
int divisor_sum(int n);
}
class Calculator implements AdvancedArithmetic {
List<Integer> l1 = new ArrayList<>();
int sum = 0;
@Override
public int divisor_sum(int n) {
for (int i=1;i<=n;i++) {
if (n%i==0) {
l1.add(i);
}
}
for (int j:l1) {
sum+=j;
}
return sum;
}
}
public class MyCalculator{
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
int num = sc.nextInt();
Calculator m1 =new Calculator();
System.out.println(m1.divisor_sum(num));
}
}
```

QUESTION – 2

```
package javatest2;
import java.util.*;
public class Alien {
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
int N = sc.nextInt();
List<String> arr1 = new ArrayList<>();
List<String> arr2 = new ArrayList<>();
for (int i=0;i<N;i++) {
arr1.add(sc.next());
}
String order = sc.next();
for (int i=0;i<N;i++) {
arr2.add(String.valueOf((arr1.get(i)).charAt(0)));
}
if (arr1==arr2) {
System.out.println(1);
}
else System.out.println(0);
}
}
```

QUESTION – 3

```
package javatest2;
import java.util.*;


class MenuItems{
public int bill(List<Integer> trio) {
int sum=0;
Collections.sort(trio);
trio.remove(0);
for (int i=0;i<trio.size();i++) {
sum+=trio.get(i);
}
return sum;
}
}

public class Menu {
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
List<Integer> trio = new ArrayList<>();
for (int i=0;i<3;i++){
trio.add(sc.nextInt());
}
MenuItems m1 = new MenuItems();
System.out.println(m1.bill(trio));
}
}
```

QUESTION – 4

```
package javatest2;
import java.util.*;


interface DigitalTree{
int absorbSunlight(int hours);
int getTreeDetails();
}


class BinaryTree implements DigitalTree{

public int absorbSunlight(int a) {

return a*2;
}


@Override
public int getTreeDetails() {
// TODO Auto-generated method stub
return 0;
}
}


class QuantumTree implements DigitalTree{
public int absorbSunlight(int b) {
return 3*(b*b);
}


@Override
public int getTreeDetails() {
// TODO Auto-generated method stub
return 0;
```

```
}
}


class NeuralTree implements DigitalTree{
public int absorbSunlight(int c) {
return c*c*c;
}


@Override
public int getTreeDetails() {
// TODO Auto-generated method stub
return 0;
}
}


class ForestManager {
static BinaryTree b1 = new BinaryTree();
static QuantumTree t1 = new QuantumTree();
static NeuralTree n1 = new NeuralTree();
static int n,e=0,TE;
List<String> types;
public int produceEnergyForForest(int n, int TE,
List<String> types) {
for (int i=0;i<n;i++) {
if (types.get(i).charAt(0)=='B') {
e+=b1.absorbSunlight(TE);
}
else if(types.get(i).charAt(0)=='Q') {
e+=t1.absorbSunlight(TE);
}
else e+=n1.absorbSunlight(TE);
}
return e;
}
```

```
public void getReport() {

}
}
public class Forest100 {
public static void main(String[] args) {

Scanner sc = new Scanner(System.in);
int n = sc.nextInt(), TE = sc.nextInt();
List<String> types = new ArrayList<>();
for (int i=0;i<n;i++) {
types.add(sc.next());
}
ForestManager f1 = new ForestManager();
System.out.println(f1.produceEnergyForForest(n,TE,types));
f1.getReport();
}
}
```