

```
1 package test2;
2
3 interface AdvancedArithmetic
4 {
5     void divisor_sum(int n);
6 }
7 class Mycalculator implements AdvancedArithmetic
8 {
9
10     @Override
11     public void divisor_sum(int n) {
12         // TODO Auto-generated method stub
13         int sum=0;
14         for( int i=1; i <= n; i++)
15             {
16                 if( n % i == 0)
17                     {
18                         sum= sum+ i;
19                     }
20             }
21         System.out.println(sum);
22     }
23 }
24 public class Q1 {
25
26     public static void main(String[] args) {
27         // TODO Auto-generated method stub
28         Mycalculator mc= new Mycalculator();
29         mc.divisor_sum(10);
30     }
31 }
32 }
33 }
```

```
1 package test2;
2
3 import java.util.ArrayList;
4 import java.util.Scanner;
5 import java.util.Stack;
6
7
8 public class Q2 {
9     public static void add(ArrayList<Integer>[] adj, char u, char v) {
10
11         adj[u - 'a'].add(v - 'a');
12     }
13     public static void Sort(ArrayList<Integer>[] adj, int u, boolean[] visited,
14         Stack<Integer> st) {
15         visited[u] = true;
16         for (int i = 0; i < adj[u].size(); i++) {
17             int v = adj[u].get(i);
18             if (!visited[v]) {
19                 Sort(adj, v, visited, st);
20             }
21         }
22         st.push(u);
23     }
24
25     public static void topSort(ArrayList<Integer>[] adj, int V) {
26
27         boolean[] visited = new boolean[V];
28         Stack<Integer> st = new Stack<Integer>();
29
30         for (int i = 0; i < V; i++) {
31             visited[i] = false;
32         }
33         for (int i = 0; i < V; i++) {
34
35             if (!visited[i]) {
36                 Sort(adj, i, visited, st);
37             }
38         }
39         while (!st.empty()) {
40
41             System.out.print((char) (st.pop() + 'a') + " ");
42         }
43
44         System.out.println("");
45         System.out.println(0);
46     }
47
48     public static void printOrder(String[] words, int n, int k) {
49
50         ArrayList<Integer>[] adj = new ArrayList[k];
51         for (int i = 0; i < k; i++) {
52             adj[i] = new ArrayList<Integer>();
53         }
54         for (int i = 0; i < n - 1; i++) {
55             String word1 = words[i];
56             String word2 = words[i + 1];
57
58             int j = 0;
```

```
1 package test2;
2
3 import java.util.Scanner;
4
5 interface Menu
6 {
7     void getnameandprice();
8 }
9 class Sandwich implements Menu
10 {
11     String sandwich1= "Chicken Sandwich";
12     String sandwich2= "Veg Sandwich";
13     int sandwichp1= 250;
14     int snadwichp2= 150;
15
16
17
18     @Override
19     public void getnameandprice() {
20         // TODO Auto-generated method stub
21     }
22 }
23
24 }
25 class Salad implements Menu
26 {
27     String salad1= "Greek salad";
28     String salad2= "Fruit Salad";
29     int saladp1= 200;
30     int saladp2= 100;
31
32     @Override
33     public void getnameandprice() {
34         // TODO Auto-generated method stub
35     }
36
37 }
38
39
40 }
41 class Drink implements Menu
42 {
43     String drink1 = "Iced Tea";
44     String drink2 = "Soda";
45     int drinkp1= 150;
46     int drinkp2= 50;
47
48     @Override
49     public void getnameandprice() {
50         // TODO Auto-generated method stub
51     }
52 }
53 class Trio implements Menu
54 {
55
56     @Override
57     public void getnameandprice()
58         // TODO Auto-generated method stub
59     {
```

```
1 package test2;
2 interface DigitalTree
3 {
4     void absorbSunlight(int n);
5     void getTreedetails();
6 }
7
8 class BinaryTree implements DigitalTree
9 {
10
11
12     @Override
13     public void absorbSunlight(int n) {
14         // TODO Auto-generated method stub
15     }
16
17     @Override
18     public void getTreedetails() {
19         // TODO Auto-generated method stub
20     }
21
22     void calc(int h)
23     {
24         int e;
25         e= h*h;
26         System.out.println(e);
27     }
28 }
29
30 }
31
32 class QuantumTree implements DigitalTree
33 {
34
35
36     @Override
37     public void absorbSunlight(int n) {
38         // TODO Auto-generated method stub
39     }
40
41     @Override
42     public void getTreedetails() {
43         // TODO Auto-generated method stub
44     }
45
46     void calc1(int h)
47     {
48         int e1= 3 * (h ^ 2) ;
49         System.out.println(e1);
50     }
51
52 }
53
54 }
55 class NeutralTree implements DigitalTree
56 {
57
58     @Override
59     public void absorbSunlight(int n) {
```