```
 1
 2 /*  DFS Depth Fist Search to find Connected Components of a
   graph
 5
 6
 7 import java.util.*;
 8
 9 class conCompDFS{
10     static Node[] G;
11     static int M;
12     static int N;
13     static int cnt = 0;
14     static int conCompCnt=0;
15     static Stack<Integer> s = new Stack<Integer>();
16
17     static class Node {
18         List<Edge> adj;
19         int n;
20         public boolean visited;
21
22         public Node(int N){
23             adj = new ArrayList<Edge>();
24             n=N;
25             visited = false;
26         }
27
28
29     }
30
31     static class Edge{
32         int to, weight;
33         public Edge(int t, int w){
34             to=t;
35             weight = w;
36         }
37     }
38
39     public static void makeGraph(int n){
40     G = new Node[n];
41         for(int i =0; i<n; i++){
```

```
42              G[i]=new Node(i);
43          }
44      }
45
46      public static void addEdge(int u,int v, int w){
47          G[u].adj.add(new Edge(v,w));
48          G[v].adj.add(new Edge(u,w));
49      }
50      public static int charN(char c){
51          return c;
52      }
53      public static void dfs(int n){
54
55          if(G[n].visited){
56              return;
57          }
58
59          G[n].visited = true;
60          s.push(n);
61
62          cnt++;
63
64          for(Edge e : G[n].adj)
65          {
66              dfs(e.to);
67          }
68
69      }
70      public static int conComp(Node[] g){
71              for(int i = 0; i<g.length;i++){
72                  if(!(g[i].visited)){
73                      conCompCnt++;
74                      dfs(i);
75                  }
76              }
77          return conCompCnt;
78      }
79
80      public static void main(String[] args){
81          Scanner scan = new Scanner(System.in);
```

```
 82
 83          int K = Integer.parseInt(scan.nextLine());
 84          int u = -1;
 85          int v = -1;
 86//        String t = scan.nextLine();
 87//        String q = scan.nextLine();
 88
 89          String temp = "String";
 90          for(int k =1; k<K; k++){
 91              temp = scan.nextLine();
 92              N = charN(temp.charAt(0))-65+1;
 93//            System.out.println(N);
 94              makeGraph(N);
 95
 96              while((temp = scan.nextLine())!=null){
 97                  if(temp.isEmpty()){
 98                      break;
 99                  }
100
101                   u = charN(temp.charAt(0))-65;
102                   v = charN(temp.charAt(1))-65;
103//                            System.out.println(u);
104//                            System.out.println(v);
105
106                   addEdge(u,v,1);
107
108//                System.out.println(k);
109             }
110             conComp(G);
111             if(k>1)
112                 System.out.println();
113             System.out.println(conCompCnt);
114             conCompCnt = 0;
115
116
117
118//           for(int z =0; z<N; z++){
119//               System.out.println(G[z].adj.size());
120//           }
121//           System.out.println(s);
```

```
122
123             scan.close();
124         }
125     }
126}
127
```