有一个数组 a[N]顺序存放 0-N，要求没隔两个数删掉一个数，到末尾时循环至开头继续进行，求最后一个被删掉的数的原始下标位置。以 8 个数(N=7)为例:{0，1，2，3，4，5，6，7}，0->1->2(删除)->3->4->5(删除)->6->7->0(删除),如此循环直到最后一个数被删除。

```
1   import java.util.Scanner;
1   public class Main {
2     public static void main(String[] args) {
3       Scanner scan =  new Scanner(System.in);
4       while (scan.hasNext()) {
5         int n = scan.nextInt();
6         boolean[] delete =  new boolean[n];
7         int count =  0;
8         int index =  0;
9         int num =  0;
10        while (count < n) {
11          for (int i =  0; i != n; ++i) {
12            if (delete[i] ==  false) {
13              ++num;
14              if (num ==  3) {
15                delete[i] =  true;
16                num =  0;
17                ++count;
18                index = i;
19              }
20            }
21          }
22        }
```

```
23          System.out.println(index);
24      }
25    }
26  }
```

输入一个字符串，求出该字符串包含的字符集合

```
1   import java.util.HashSet;
2   import java.util.Scanner;
3   import java.util.Set;
4   public class Main{
5           public static void main(String[] args){
6                   Scanner in =  new Scanner(System.in);
7                   while(in.hasNext()){
8                           char[] c = in.next().toCharArray();
9                           StringBuffer sb =  new StringBuffer();
10                          Set<Character> set =  new HashSet<Character>();
11                          for(int i =  0;i<c.length;i++){
12                                  if(set.add(c[i]))
13                                          sb.append(c[i]);
14                          }
15                          System.out.println(sb.toString());
16                  }
17          }
18  }
```

数独是一个我们都非常熟悉的经典游戏，运用计算机我们可以很快地解开数独难题，现在有一些简单的数独题目，请编写一个程序求解。

```
1
2
3   #include <iostream>
4   #include <vector>
5   #include <set>
6   #define REP(i,n) for(int i=0;i<(n);i++)
7   #define FOR(i,a,b) for(int i=(a);i<(b);i++)
8   using namespace std;
9   int Matrix[9][9];
10  int RE[9][9];
11  int x[81];
12  int y[81];
13  int sum0;
14  bool shuchu=false;
15  void dfs(int i){
16          if(i==sum0){
17                  if(!shuchu){
18                          REP(i1,9)REP(i2,9) RE[i1][i2]=Matrix[i1][i2];
19                         shuchu=true;
20                  }
21                  return;
22          }
23          else{
24                  set<int> st;
```

```
25                  REP(t,9){st.insert(Matrix[x[i]][t]);st.insert(Matrix[t][y[i]]);}
26                  FOR(t,1,10){
27                          if(st.count(t)!=0)continue;
28                          Matrix[x[i]][y[i]]=t;
29                          dfs(i+1);
30                  }
31                  Matrix[x[i]][y[i]]=0;
32          }
33  }
34  int main(){
35          while(cin>>Matrix[0][0]){
36                  FOR(i,1,9) cin>>Matrix[0][i];
37                  FOR(i,1,9) REP(j,9) cin>>Matrix[i][j];
38                  REP(i,9)REP(j,9) RE[i][j]=Matrix[i][j];
39                  sum0=0;
40                  REP(i,9) REP(j,9)  if(Matrix[i][j]==0) {x[sum0]=i;y[sum0]=j;sum0++;}
41                  dfs(0);
42                  REP(i,9){ REP(j,9) {cout<<RE[i][j];  if(j!=8) cout<<" ";} cout<<endl;}
43
44          }
45  }
```