```
#define  M 5010                  //题目中可能的最大点数
int STACK[M],top=0;               //Tarjan 算法中的栈
bool InStack[M];                  //检查是否在栈中
int DFN[M];                       //深度优先搜索访问次序
int Low[M];                       //能追溯到的最早的次序
int ComponentNumber=0;            //有向图强连通分量个数
int Index=0;                      //索引号
vector <int> Edge[M];             //邻接表表示
vector <int> Component[M];        //获得强连通分量结果
int InComponent[M];    //记录每个点在第几号强连通分量里
int ComponentDegree[M];           //记录每个强连通分量的度
void Tarjan(int i)
{
    int j;
    DFN[i]=Low[i]=Index++;
    InStack[i]=true;
    STACK[++top]=i;
    for (int e=0;e<Edge[i].size();e++)
    {
        j=Edge[i][e];
        if (DFN[j]==-1)
        {
            Tarjan(j);
            Low[i]=min(Low[i],Low[j]);
        }
        else if (InStack[j])
            Low[i]=min(Low[i],DFN[j]);
    }
    if (DFN[i]==Low[i])
    {
        ComponentNumber++;
        do
        {
            j=STACK[top--];
            InStack[j]=false;
            Component[ComponentNumber].push_back(j);
           InComponent[j]=ComponentNumber;
        }
        while (j!=i);
    }
}
```

```
void solve(int N)     //N 是此图中点的个数，注意是 0-indexed!
{
    memset(STACK,-1,sizeof(STACK));
    memset(InStack,0,sizeof(InStack));
    memset(DFN,-1,sizeof(DFN));
    memset(Low,-1,sizeof(Low));

    for(int i=0;i<N;i++)
        if(DFN[i]==-1)
            Tarjan(i);
}
```