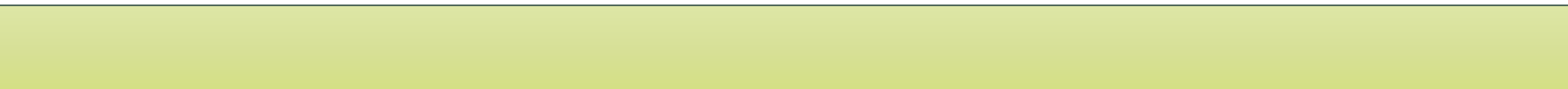




# BUGLIFE - A Bug's Life(SPOJ)

Presented by: Adarsh Singh



# Question

- <http://www.spoj.com/problems/BUGLIFE/>

Professor Hopper is researching the sexual behavior of a rare species of bugs. He assumes that they feature two different genders and that they only interact with bugs of the opposite gender. In his experiment, individual bugs and their interactions were easy to identify, because numbers were printed on their backs.

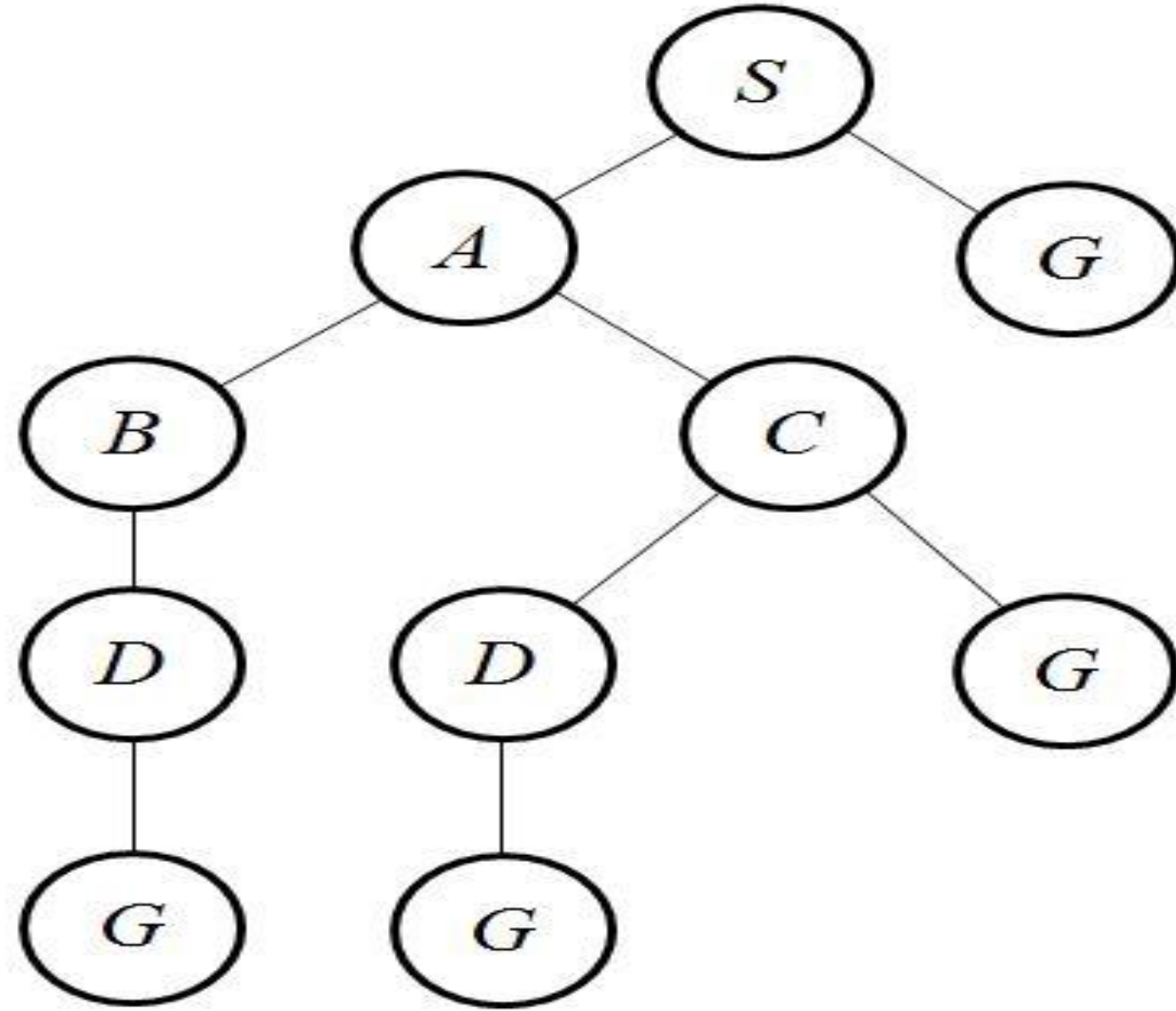
Given a list of bug interactions, decide whether the experiment supports his assumption of two genders with no homosexual bugs or if it contains some bug interactions that falsify it.

# Input Format

The first line of the input contains the number of scenarios. Each scenario starts with one line giving the number of bugs (at least one, and up to 2000) and the number of interactions (up to 1000000) separated by a single space. In the following lines, each interaction is given in the form of two distinct bug numbers separated by a single space. Bugs are numbered consecutively starting from one.

# Output Format

The output for every scenario is a line containing “Scenario #i:”, where i is the number of the scenario starting at 1, followed by one line saying either “No suspicious bugs found!” if the experiment is consistent with his assumption about the bugs’ sexual behavior, or “Suspicious bugs found!” if Professor Hopper’s assumption is definitely wrong.





# Solution using BFS

adja.cpp

x

```
1  #include<bits/stdc++.h>
2
3  using namespace std;
4
5  vector< vector<int> >v;    // vector for maintaining adjacency list explained above.
6  int level[3000]={0};    // to determine the level of each node
7  bool vis[3000];        //mark the node if visited
8  int lol[1000005][2];
9  void bfs(int s) {
10     queue <int> q;
11     q.push(s);
12     level[ s ] = 0 ;    //setting the level of sources node as 0.
13     vis[ s ] = true;
14     while(!q.empty())
15     {
16         int p = q.front();
17         q.pop();
18         for(int i = 0;i < v[ p ].size() ; i++)
19         {
20             if(vis[ v[ p ][ i ] ] == false)
21             {
22                 //setting the level of each node with an increment in the level of parent node
23                 level[ v[ p ][ i ] ] = level[ p ]+1;
24                 q.push(v[ p ][ i ]);
25                 vis[ v[ p ][ i ] ] = true;
26             }
27         }
28     }
29 }
30
```

```
30
31 void init(int n)
32 {
33     for(int i=0;i<n+5;i++)
34         vis[i]=false;
35 }
36
37 int main()
38 {
39     int t;
40     cin>>t;
41     int k=0;
42     while(t--){
43         int nodes,edges,x,y;
44         cin>>nodes>>edges;
45         v.resize(nodes+5);
46         v.clear();
47         int i;
48         for(i=0;i<edges;i++)
49         {
50             cin>>x>>y;
51             v[x].push_back(y);
52             v[y].push_back(x);
53             lol[i][0]=x;
54             lol[i][1]=y;
55         }
56         init(nodes);
57         for(i=1;i<=nodes;i++)
58         {
59             if(vis[i]==false)
```



adja.cpp

x

```
55     }
56     init(nodes);
57     for(i=1;i<=nodes;i++)
58     {
59         if(vis[i]==false)
60             bfs(i);
61     }
62     //bfs(lol[0][0]);
63     int flag=0;
64     for(i=0;i<edges;i++)
65     {
66         if((level[lol[i][0]]-level[lol[i][1]])%2 == 0)
67         {
68             flag=1;
69             break;
70         }
71     }
72
73     k++;
74     if(flag==1)
75     {
76         cout<<"Scenario #"<<k<<":\nSuspicious bugs found!\n";
77     }
78     else
79     {
80         cout<<"Scenario #"<<k<<":\nNo suspicious bugs found!\n";
81     }
82 }
83 return 0;
84 }
```



Thanks