



Practice

Compete

Jobs

Rank

Leaderboard



bor0s ▾

Dashboard > Algorithms > Graph Theory > Even Tree

Badge Progress



Points: 371.00 Rank: 88214

Even Tree



by HackerRank

Problem

Submissions

Leaderboard

Discussions

Editorial

You are given a tree (a simple connected graph with no cycles). The tree has N nodes numbered from 1 to N and is rooted at node 1 .

Find the maximum number of edges you can remove from the tree to get a **forest** such that each connected component of the forest contains an even number of vertices.

Input Format

The first line of input contains two integers N and M . N is the number of vertices, and M is the number of edges.

The next M lines contain two integers u_i and v_i which specifies an edge of the tree.

Constraints

- $2 \leq N \leq 100$

Note: The tree in the input will be such that it can always be decomposed into components containing an even number of nodes.

Output Format

Print the number of removed edges.

Sample Input

```
10 9
2 1
3 1
4 3
5 2
6 1
7 2
8 6
9 8
10 8
```

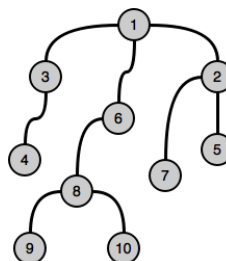
Sample Output

```
2
```

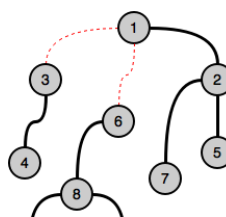
Explanation

On removing edges $(1, 3)$ and $(1, 6)$, we can get the desired result.

Original tree:



Decomposed tree:





[f](#) [t](#) [in](#)Submissions: [20141](#)

Max Score: 50

Difficulty: Medium

Rate This Challenge:

[More](#)Current Buffer (saved locally, editable)  

C++14



```

1 #include <cmath>
2 #include <cstdio>
3 #include <vector>
4 #include <iostream>
5 #include <algorithm>
6 using namespace std;
7
8 // #define DEBUG
9
10 int edge_count(vector<int> *g, int edge) {
11     if (g[edge].size() == 0) {
12         return 0;
13     } else {
14         int sum = 0;
15
16         for (int i = 0; i < g[edge].size(); i++) {
17             sum += 1 + edge_count(g, g[edge][i]);
18         }
19
20         return sum;
21     }
22 }
23
24 void print_graph(vector<int> *g, int m) {
25     for (int i = 1; i <= m; i++) {
26         if (g[i].size() == 0) {
27             continue;
28         }
29
30         cout << i << ": ";
31
32         for (int j = 0; j < g[i].size(); j++) {
33             cout << g[i][j] << " ";
34         }
35
36         cout << "(count: " << 1 + edge_count(g, i) << ")\n";
37     }
38 }
39
40 int max_removals(vector<int> *g, int edge) {
41     if (g[edge].size() == 0) {
42         return 0;
43     } else {
44         int removals = 0;
45
46         for (int i = 0; i < g[edge].size(); i++) {
47             int count = 1 + edge_count(g, g[edge][i]);
48             removals += (count % 2 == 0 ? 1 : 0) + max_removals(g, g[edge][i]);
49 #ifdef DEBUG
50             if (count % 2 == 0) {
51                 cout << "remove edge " << g[edge][i] << ", " << edge << "\n";
52             }
53 #endif
54         }
55
56         return removals;
57     }
58 }
59
60 void read_graph(vector<int> *g, int m) {
61     for (int i = 0; i < m; i++) {
62         int edgeA, edgeB;
63         cin >> edgeA >> edgeB;
64         if (g[edgeA].size() > 0) g[edgeA].push_back(edgeB);
65         else g[edgeB].push_back(edgeA);
66     }
67 }
68
69
70 int main() {
71     int n, m, removals = 0;
72
73     cin >> n >> m;
74     vector<int> graph[n + 1];
75
76     read_graph(graph, m);
77
78     cout << max_removals(graph, 1) << "\n";
79 #ifdef DEBUG
80     print_graph(graph, m);
81 #endif
82     return 0;
83 }
84

```

Line: 1 Col: 1

☒ Upload Code as File
 ☐ Test against custom input

Run Code

Submit Code

Copyright © 2017 HackerRank. All Rights Reserved

Join us on IRC at [#hackerrank](#) on freenode for hugs or bugs.

[Contest Calendar](#) | [Interview Prep](#) | [Blog](#) | [Scoring](#) | [Environment](#) | [FAQ](#) | [About Us](#) | [Support](#) | [Careers](#) | [Terms Of Service](#) | [Privacy Policy](#) | [Request a Feature](#)