Worksheet-2.3

Student Name:- Pushpraj Roy **UID:-** 20BCS9866

Branch:- BE- CSE **Section/Group:-** WM_617 “A”

Subjetct Code:- 20CSP-314 **Semester:-** 5th

Subject Name:- Competitive Coding Lab

Problem 1:- Journey-to-the-moon

https://www.hackerrank.com/challenges/journey-to-the-moon/problem?isFullScreen=true

Code:-

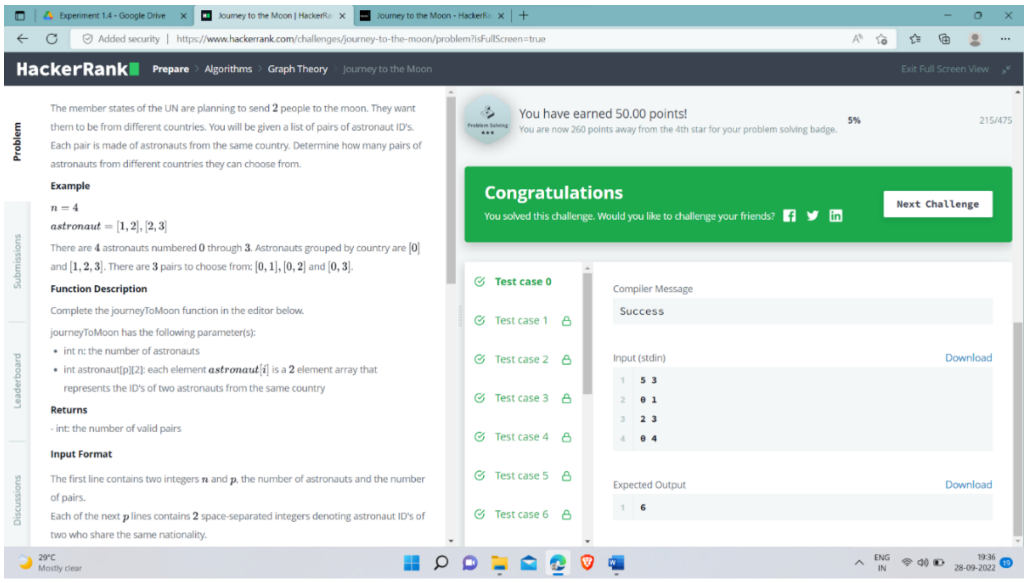
```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include <map>
using namespace std;

vector<int> parent;
vector<int> rankk;
vector<int> v;

int find_set (int v) {
    if (v == parent[v])
            return v;
    return parent[v] = find_set (parent[v]);
}
```

```
void union_sets (int a, int b) {
    a = find_set (a);
    b = find_set (b);
    if (a != b) {
        if (rankk[a] < rankk[b])
            swap (a, b);
        parent[b] = a;
        if (rankk[a] == rankk[b])
            ++rankk[a];
    }
}
int n, m;
map<int,int> mm;
int main() {
    cin >> n >> m;
    parent.resize(n);
    rankk.resize(n);
    for (int i = 0; i != n; ++i)
    {
        parent[i] = i;
        rankk[i] = 0;
    }
    for (int i = 0; i != m; ++i)
    {
        int x,y;
        cin >> x >> y;
        union_sets(x,y);
    }
    for (int i = 0; i != n; ++i)
    {
        mm[find_set(i)]++;
        }
        map<int,int>::iterator it = mm.begin();
        map<int,int>::iterator itEnd = mm.end();
        long long res = 0;
        int b = 0;
        for (; it != itEnd; ++it)
        {
                v.push_back(it->second);
        }
        int l = v.size();
        long long rr = 0;
        for (int i = 0; i != l; ++i)
                rr += v[i];
        for (int i = 0; i != l; ++i)
        {
                rr -= v[i];
                res += v[i]*rr;
        }
        cout << res << endl;
        return 0;
}
```

Output:-



Experiment 1.4 - Google Drive
Journey to the Moon | HackerRank
Journey to the Moon - HackerRank
Added security | https://www.hackerrank.com/challenges/journey-to-the-moon/problem?isFullScreen=true
HackerRank
Prepare > Algorithms > Graph Theory > Journey to the Moon
Exit Full Screen View
Problem
Submissions
Leaderboard
Discussions

The member states of the UN are planning to send 2 people to the moon. They want them to be from different countries. You will be given a list of pairs of astronaut ID's. Each pair is made of astronauts from the same country. Determine how many pairs of astronauts from different countries they can choose from.

Example

$n = 4$

$astronaut = [1, 2], [2, 3]$

There are 4 astronauts numbered 0 through 3. Astronauts grouped by country are [0] and [1, 2, 3]. There are 3 pairs to choose from: [0, 1], [0, 2] and [0, 3].

Function Description

Complete the journeyToMoon function in the editor below.

journeyToMoon has the following parameter(s):

- int n: the number of astronauts
- int astronaut[p][2]: each element $astronaut[i]$ is a 2 element array that represents the ID's of two astronauts from the same country

Returns

- int: the number of valid pairs

Input Format

The first line contains two integers n and p, the number of astronauts and the number of pairs.

Each of the next p lines contains 2 space-separated integers denoting astronaut ID's of two who share the same nationality.

You have earned 50.00 points!
You are now 260 points away from the 4th star for your problem solving badge.
5% 215/475

Congratulations
You solved this challenge. Would you like to challenge your friends?

Compiler Message
```
Success
```
Input (stdin)
```
5 3
0 1
2 3
0 4
```
Expected Output
```
6
```