

牛客多校第二场

这场彻底跪了。

那我贴一贴莫名WA掉的代码们吧。如果读者能找到到底是哪里WA掉了，麻烦您联系我一下。

C

我觉得从奇顶点开始DFS一定没错。但是评测机不这么认为.....

```
#include<stdio.h>
#include<vector>

using namespace std;

int degree[200010];

vector<int> Adj[200010];
bool vis[200010] = {false};

int flag;

void DFS(int u)
{
    vis[u]=true;
    if(degree[u]%2==1)
    {
        printf("%d",u);
        (flag==0)?putchar(' '):putchar('\n');
        flag^=1;
    }
    int i;
    for(i=0;i<Adj[u].size();i++)
    {
        int v=Adj[u][i];
        if(vis[v]==false)
        {
            DFS(v);
        }
    }
}

int main()
{
    int n;
    scanf("%d",&n);
    if(n==1)
```

```
{
    printf("1\n1 1\n");
    return 0;
}
int m=n-1;
int x,y;
int sum=0;
while(m-->0)
{
    scanf("%d%d",&x,&y);
    degree[x]++;
    degree[y]++;
    Adj[x].push_back(y);
    Adj[y].push_back(x);
    if(degree[x]%2==1&&degree[y]%2==1)
    {
        sum++;
    }
    else if(degree[x]%2==0&&degree[y]%2==0)
    {
        sum--;
    }
}
printf("%d\n",sum);
flag=0;
int u;
for(u=0;u<n;u++)
{
    if(vis[u]==false&&degree[u]%2==1)
    {
        DFS(u);
        break;
    }
}
}
```

后记：我知道错到哪里了！我一直以为是不重复的覆盖.....

其实重复也是可以的，也就是说我硬生生拔高了原题的难度.....

唉，怪不得。那么代码就更简单了low的不谈

上面这个代码是不重复覆盖的优秀代码

B

储存斜率用了两种做法。开longlong那个TLE掉了double那个WA了，实在无语.....

```
#include<iostream>
#include<vector>
#include<map>

using namespace std;

#define x first
#define y second

vector<pair<double,double> > points;

int maxPoints()
{
    int res = 0;
    for (int i = 0; i < points.size(); ++i)
    {
        map<double,int> m;
        int duplicate = 1;
        for (int j = i + 1; j < points.size(); ++j)
        {
            if (points[i].x== points[j].x&& points[i].y== points[j].y)
            {
                ++duplicate;
                continue;
            }
            if(points[j].x*points[i].y==points[i].x*points[j].y)
            {
                continue;
            }
            double dx = points[j].x- points[i].x;
            double dy = points[j].y- points[i].y;
            ++m[{dx/dy}];
        }
        res = max(res, duplicate);
        map<double,int>::iterator it;
        for (it = m.begin(); it != m.end(); ++it)
        {
            res = max(res, it->second + duplicate);
        }
    }
    return res;
}

int main()
{
    int n;
    scanf ("%d",&n);
    double x,y;
    while(n--)
    {
```

```
    cin >> x >> y;  
    double temp=x*x+y*y;  
    points.push_back(make_pair(x/temp,y/temp));  
}  
cout << maxPoints() << endl;  
return 0 ;  
}
```

```
#include<iostream>  
#include<vector>  
#include<map>  
  
using namespace std;  
  
vector<pair<long long,long long> > points;  
  
long long gcd(long long a,long long b)  
{  
    return (b == 0) ? a : gcd(b, a % b);  
}  
  
int maxPoints()  
{  
    int res = 0;  
    for (int i = 0; i < points.size(); ++i)  
    {  
        map<pair<int, int>, int> m;  
        int duplicate = 1;  
        for (int j = i + 1; j < points.size(); ++j)  
        {  
            if (points[i].first== points[j].first&& points[i].second==  
points[j].second)  
            {  
                ++duplicate;  
                continue;  
            }  
            long long x1=points[i].first;  
            long long x2=points[j].first;  
            long long y1=points[i].second;  
            long long y2=points[j].second;  
            if(x1*y2==y1*x2)  
            {  
                continue;  
            }  
            long long dx =x2*(x1*x1+y1*y1)-x1*(x2*x2+y2*y2);  
            long long dy =y2*(x1*x1+y1*y1)-y1*(x2*x2+y2*y2);  
            long long d = gcd(dx, dy);  
            ++m[{dx / d, dy / d}];  
        }  
    }  
    return res;  
}
```

```
    }
    res = max(res, duplicate);
    map<pair<int,int>,int>::iterator it;
    for(it = m.begin(); it != m.end(); ++it)
    {
        res = max(res, it->second + duplicate);
    }
}
return res;
}

int main()
{
    int n;
    scanf("%d",&n);
    long long x,y;
    while(n--)
    {
        cin >> x >> y;
        points.push_back(make_pair(x,y));
    }
    cout << maxPoints() << endl;
    return 0 ;
}
```

后记：这种采用了反演的写法要考虑eps

From:
<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:
<https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:namespace:%E7%89%9B%E5%AE%A2%E5%A4%9A%E6%A0%A1%E7%AC%AC%E4%BA%8C%E5%9C%BA&rev=1594730196>

Last update: 2020/07/14 20:36