最小生成树应用

1. P1536村村通

- **题目描述**
- 某市调查城镇交通状况，得到现有城镇道路统计表。表中列出了每条道路直接连通的城镇。市政府“村村通工程”的目标是使全市任何两个城镇间都可以实现交通（但不一定有直接的道路相连，只要相互之间可达即可）。请你计算出最少还需要建设多少条道路？
- **输入格式**
- 每个输入文件包含若干组测试数据，每组测试数据的第一行给出两个用空格隔开的正整数，分别是城镇数目**N**（**N<1000**）和道路数目**M**；随后的**M**行对应**M**条道路，每行给出一对用空格隔开的正整数，分别是该条道路直接相连的两个城镇的编号。简单起见，城镇从**1**到**N**编号。
- 注意：两个城市间可以有多条道路相通。例如：
- **3 3 1 2 1 2 2 1** 这组数据也是合法的。当**N**为**0**时，输入结束。
- **输出格式**
- 对于每组数据，对应一行一个整数。表示最少还需要建设的道路数目。

输入输出样例

输入 #1	输出 #1
4 2 1 3 4 3 3 3 1 2 1 3 2 3 5 2 1 2 3 5 999 0 0	1 0 2 998

2. P2504 [HAOI2006]聪明的猴子

题目描述

在一个热带雨林中生存着一群猴子，它们以树上的果子为生。昨天下了一场大雨，现在雨过天晴，但整个雨林的地表还是被大水淹没着，部分植物的树冠露在水面上。猴子不会游泳，但跳跃能力比较强，它们仍然可以在露出水面的不同树冠上来回穿梭，以找到喜欢吃的果实。

现在，在这个地区露出水面的有N棵树，假设每棵树本身的直径都很小，可以忽略不计。我们在这块区域上建立直角坐标系，则每一棵树的位置由其所对应的坐标表示(任意两棵树的坐标都不相同)。

在这个地区住着的猴子有M个，下雨时，它们都躲到了茂密高大的树冠中，没有被大水冲走。由于各个猴子的年龄不同、身体素质不同，它们跳跃的能力不同。有的猴子跳跃的距离比较远(当然也可以跳到较近的树上)，而有些猴子跳跃的距离就比较近。这些猴子非常聪明，它们通过目测就可以准确地判断出自己能否跳到对面的树上。

【问题】现已知猴子的数量及每一个猴子的最大跳跃距离，还知道露出水面的每一棵树的坐标，你的任务是统计有多少个猴子可以在这个地区露出水面的所有树冠上觅食。

2. P2504 [HAOI2006]聪明的猴子

输入格式

输入文件**MONKEY.IN**包括：

第**1**行为一个整数，表示猴子的个数**M(2<=M<=500)**；

第**2**行为**M**个整数，依次表示猴子的最大跳跃距离（每个整数值在**1--1000**之间）；

第**3**行为一个整数表示树的总棵数**N(2<=N<=1000)**；

第**4**行至第**N+3**行为**N**棵树的坐标（横纵坐标均为整数，范围为：**-1000--1000**）。

（同一行的整数间用空格分开）

输出格式

输出文件**MONKEY.OUT**包括一个整数，表示可以在这个地区的所有树冠上觅食的猴子数。

2. P2504 [HAOI2006]聪明的猴子

输入输出样例

输入 #1 复制

```
4
 1 2 3 4
6
0 0
1 0
1 2
-1 -1
-2 0
2 2
```

输出 #1 复制

```
3
```

说明/提示

【数据规模】

对于40%的数据，保证有2<=N <=100，1<=M<=100

对于全部的数据，保证有2<=N <= 1000，1<=M=500

3. P2872 [USACO07DEC]道路建设BUILDING ROADS

问题描述：

FARMER JOHN最近得到了一些新的农场，他想新修一些道路使得他的所有农场可以经过原有的或是新修的道路互达（也就是说，从任一个农场都可以经过一些首尾相连道路到达剩下的所有农场）。有些农场之间原本就有道路相连。 所有**N(1 <= N <= 1,000)**个农场（用**1..N**顺次编号）在地图上都表示为坐标为**(X_I, Y_I)**的点**(0 <= X_I <= 1,000,000； 0 <= Y_I <= 1,000,000)**，两个农场间道路的长度自然就是代表它们的点之间的距离。现在**FARMER JOHN**也告诉了你农场间原有的**M(1 <= M <= 1,000)**条路分别连接了哪两个农场，他希望你计算一下，为了使得所有农场连通，他所需建造道路的最小总长是多少。

3. P2872 [USACO07DEC]道路建设BUILDING ROADS

- 输入格式
- 第1行:2个用空格隔开的整数：N 和 M
- 第2..N+1行:第I+1行为2个用空格隔开的整数：X_I、Y_I
- 第N+2..N+M+2行:每行用2个以空格隔开的整数I、J描述了一条已有的道路， 这条道路连接了农场I和农场J
- 输出格式
- *输出使所有农场连通所需建设道路的最小总长，保留2位小数，不必做 任何额外的取整操作。为了避免精度误差，计算农场间距离及答案时 请使用64位实型变量

3. P2872 [USACO07DEC]道路建设BUILDING ROADS

输入输出样例

输入 #1 复制

```
4 1
1 1
3 1
2 3
4 3
1 4
```

输出 #1 复制

```
4.00
```

说明/提示

题目简述：给出n个点的坐标,其中一些点已经连通,现在要把所有点连通,求修路的最小长度.

3. P2872 [USACO07DE
ROADS

```
#define  maxn 1005
typedef long long ll;
const ll inf=1e+18;
using namespace std;
ll f[maxn],n,m,a,b,num=0;
double ans=0;
ll find_father(ll x)
{
    return f[x]==x?x:f[x]=find_father(f[x]);
}
struct Node{
    ll x,y;
}node[maxn];
struct Edges
{
    ll u,v;
    double val;//u、v之间的边权val
    friend bool operator <(Edges a,Edges b)
    {
        return a.val<b.val;
    }
}edges[maxn*maxn/2+10];//注意数组大小，边数

```

```
int main()
{
    //freopen("/Users/zhangkanqi/Desktop/11.txt","r",stdin);
    scanf("%lld %lld",&n,&m);
    for(ll i=1;i<=n;i++)
    {
        f[i]=i;//初始化
        scanf("%lld %lld",&node[i].x,&node[i].y);
    }
    for(ll i=1;i<=n;i++)
    {
        for(ll j=i+1;j<=n;j++)//j=i+1
        {
            double d=sqrt(double(node[i].x-node[j].x)*(node[i].x-node[j].x)+double(
            edges[++num].u=i;
            edges[num].v=j;
            edges[num].val=d;
        }
    }
    for(ll i=1;i<=m;i++)
    {
        scanf("%lld %lld",&a,&b);
        edges[++num].u=a;
        edges[num].v=b;
        edges[num].val=0;//虽然多加了边数，但是在生成最小生成树的时候优先选，所以原来的val
    }
    sort(edges+1,edges+1+num);
    for(ll i=1;i<=num;i++)
    {
        ll x=find_father(edges[i].u);
        ll y=find_father(edges[i].v);
        if(x!=y)
        {
            f[x]=y;//合并
            ans+=edges[i].val;
        }
    }
    printf("%.2lf\n",ans);
```

4. P1195 口袋的天空

- **题目背景**

小杉坐在教室里，透过口袋一样的窗户看口袋一样的天空。

有很多云飘在那里，看起来很漂亮，小杉想摘下那样美的几朵云，做成棉花糖。

- **题目描述**

给你云朵的个数**N**，再给你**M**个关系，表示哪些云朵可以连在一起。

现在小杉要把所有云朵连成**K**个棉花糖，一个棉花糖最少要用掉一朵云，小杉想知道他怎么连，花费的代价最小。

4. P1195 口袋的天空

- **输入格式**

每组测试数据的

第一行有三个数**N,M,K(1≤N≤1000,1≤M≤10000,1≤K≤10)**

接下来**M**个数每行三个数**X,Y,L**，表示**X**云和**Y**云可以通过**L**的代价连在一起。**(1≤X,Y≤N,0≤L<10000)**

30%的数据**N≤100,M≤100000**

- **输出格式**

对每组数据输出一行，仅有一个整数，表示最小的代价。

如果怎么连都连不出**K**个棉花糖，请输出**'NO ANSWER'**。

输入输出样例

输入 #1 复制	输出 #1 复制
3 1 2 1 2 1	1

5. P1265 公路修建

- 题目描述

某国有n个城市，它们互相之间没有公路相通，因此交通十分不便。为解决这一“行路难”的问题，政府决定修建公路。修建公路的任务由各城市共同完成。

修建工程分若干轮完成。在每一轮中，每个城市选择一个与它最近的城市，申请修建通往该城市的公路。政府负责审批这些申请以决定是否同意修建。

政府审批的规则如下：

（1）如果两个或以上城市申请修建同一条公路，则让它们共同修建；

（2）如果三个或以上的城市申请修建的公路成环。如下图，A申请修建公路AB，B申请修建公路BC，C申请修建公路CA。则政府将否决其中最短的一条公路的修建申请；

（3）其他情况的申请一律同意。

5. P1265 公路修建

一轮修建结束后，可能会有若干城市可以通过公路直接或间接相连。这些可以互相连通的城市即组成“城市联盟”。在下一轮修建中，每个“城市联盟”将被看作一个城市，发挥一个城市的作用。

当所有城市被组合成一个“城市联盟”时，修建工程也就完成了。

你的任务是根据城市的分布和前面讲到的规则，计算出将要修建的公路总长度。

- **输入格式**

第一行一个整数**N**，表示城市的数量。**(N≤5000)**

以下**N**行，每行两个整数**X**和**Y**，表示一个城市的坐标。**(-1000000≤X，Y≤1000000)**

- **输出格式**

一个实数，四舍五入保留两位小数，表示公路总长。（保证有惟一解）

5. P1265 公路修建

输入输出样例

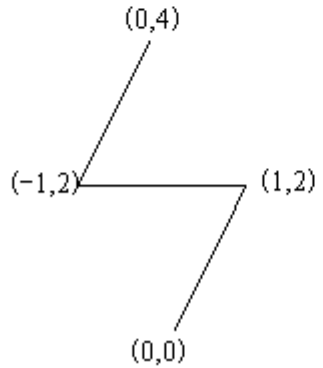
输入 #1 复制

```
4
0 0
1 2
-1 2
0 4
```

输出 #1 复制

```
6.47
```

说明/提示



修建的公路如图所示：

6. P1991 无线通讯网

- **题目描述**

国防部计划用无线网络连接若干个边防哨所。**2** 种不同的通讯技术用来搭建无线网络；

每个边防哨所都要配备无线电收发器；有一些哨所还可以增配卫星电话。

任意两个配备了一条卫星电话线路的哨所（两边都有卫星电话）均可以通话，无论他们相距多远。而只通过无线电收发器通话的哨所之间的距离不能超过 **D**，这是受收发器的功率限制。收发器的功率越高，通话距离 **D** 会更远，但同时价格也会更贵。

收发器需要统一购买和安装，所以全部哨所只能选择安装一种型号的收发器。换句话说，每一对哨所之间的通话距离都是同一个 **D**。你的任务是确定收发器必须的最小通话距离 **D**，使得每一对哨所之间至少有一条通话路径（直接的或者间接的）。

6. P1991 无线通讯网

- **输入格式**

从 **WIRELESS.IN** 中输入数据第 **1** 行，**2** 个整数 **S** 和 **P**，**S** 表示可安装的卫星电话的哨所数，**P** 表示边防哨所的数量。接下里 **P** 行，每行两个整数 **X**，**Y** 描述一个哨所的平面坐标**(X, Y)**，以 **KM** 为单位。

- **输出格式**

输出 **WIRELESS.OUT** 中

第 **1** 行，**1** 个实数 **D**，表示无线电收发器的最小传输距离，精确到小数点后两位。

6. P1991 无线通讯网

输入输出样例

输入 #1 复制

```
2 4
0 100
0 300
0 600
150 750
```

输出 #1 复制

```
212.13
```

说明/提示

对于 20% 的数据：P = 2，S = 1

对于另外 20% 的数据：P = 4，S = 2

对于 100% 的数据保证：1 ≤ S ≤ 100，S < P ≤ 500，0 ≤ x，y ≤ 10000。

次优最小生成树

- 次优最小生成树是由最小生成树而来的，含义就是所有的生成树集合中，除去最小的那棵，剩下的集合中最小的生成树。（如果所有边的权值都不同，那么次优生成树是一定大于最小生成树的，但是如果存在边的权值相同，则次优生成树可能会等于最小生成树）
- 基本思路是：先求出最小生成树**T**，然后将不属于最小生成树中的边加入**T**中，此时会形成一个环，然后将环中最大的一条边（刚加入的边不算）去掉，这样可以保证增加量是最小的。依次遍历所有的不在**T**中的边，即可求出次优生成树的值。

次优最小生成树

- 方法：用**PRIM**算法求一棵最小生成树，利用**PRIM**算法的特性，即对于每一步扩展，都保持扩展的结果是一棵树，为了方便下一步枚举边的判断，我们用一个**MAX**数组记录**I**点跟**J**点之间所有边中最大的一条边，这一步在**PRIM**算法中很容易做到，因为**MAX[I][J]=MAX{MAX[I][K],EDGE[K][J]}**。接下来的操作就是枚举每一条不在最小生成树中的边，加入**T**中，然后将环中最大边去掉。

9. POJ1679

- 题意：给定一个无向图，判断最小生成树是否唯一。
- 解析：先求出最小生成树，然后求出次优最小生成树，比较两个值是否相等。

8. POJ1679

```
#define N 105
#define INF 0xfffffff
using namespace std;
int G[N][N], used[N][N],MAX[N][N];
int pre[N], dis[N];
bool vis[N];
int m, n;
void init(){
	for(int i = 1; i <= n; i++){
		vis[i] = false;
		//dis[i] = INF;
		for(int j = 1; j <= n; j++)
			G[i][j] = INF;
	}
	memset(used, 0, sizeof(used));
	memset(path, 0, sizeof(MAX));
}
```

8. POJ1679

```
                for(int j = 1; j <= n; j++){
                    if(vis[j] && j != st)
                        MAX[j][st] = MAX[st][j] = max(MAX[j][pre[st]], dis[st]);
                    if(!vis[j] && dis[j] > G[st][j]){
                        dis[j] = G[st][j];
                        pre[j] = st;
                    }
                }
            }
            return sum;
}
int sec_Prim(int tmp){
            int sum = INF;
            for(int i = 1; i <= n; i++){
                for(int j = 1; j <= n; j++){
                    if(i != j && !used[i][j])
                        sum = min(sum, tmp + G[i][j] - MAX[i][j]);
                }
            }
            return sum;
}
```

8. POJ1679

```
int main(){
	int t;
	int st, ed, len;
	scanf("%d", &t);
	while(t--){
		scanf("%d%d", &n, &m);
		init();
		for(int i = 0; i < m; i++){
			scanf("%d%d%d", &st, &ed, &len);
			G[st][ed] = G[ed][st] = len;
		}
		int ans1 = Prim(1);
		int ans2 = sec_Prim(ans1);
		if(ans1 != ans2)
			printf("%d\n", ans1);
		else
			puts("Not Unique!");
	}
	return 0;
}
```