（山东大学附属中学 赵宗昌 2022.06.23 ）

第一部分 快速幂

1.P1226【模板】快速幂||取余运算

题目描述

给你三个整数 a,b,p，求 $a^b \bmod p$。

输入格式

输入只有一行三个整数，分别代表 a,b,p。

输出格式

输出一行一个字符串 `a^b mod p=s`，其中 a,b,p 分别为题目给定的值， s 为运算结果。

样例 #1

样例输入 #1

```
2 10 9
```

样例输出 #1

```
2^10 mod 9=7
```

提示

样例解释

$2^{10}=1024$，$1024 \bmod 9=7$。

数据规模与约定

对于 100% 的数据，保证 $0 \le a,b < 2^{31}$，$a+b>0$，$2 \le p < 2^{31}$。

参考代码

```
//分治思想：递归实现
#include<cstdio>
#include<iostream>
#define LL long  long
using namespace std;
int b,p,k;
int fast(LL a,int n,int p){
    if(n==0)return 1%p;
    if(n==1)return a%p;
    LL s=fast(a,n/2,p);
    s=(s*s)%p;
```

```
    if(n&1)s=(s*a)%p;
    return s;
}
int main(){
    cin>>b>>p>>k;
    printf("%d^%d mod %d=%d\n",b,p,k,fast(b,p,k));
    return 0;
}
```

```
//倍增思想实现
#include<cstdio>
#include<iostream>
#define LL long long
using namespace std;
int a,b,p;
LL Fast(LL a,int n,int p){
    LL s=1%p;
    while(n){
        if(n&1)s=(s*a)%p;
        a=(a*a)%p;
        n=n>>1;
    }
    return s;
}
int main(){
    scanf("%d%d%d",&a,&b,&p);
    printf("%d^%d mod %d=%lld\n",a,b,p,Fast(a,b,p));
    return 0;
}
```

2.幂的和

题目描述

输入 x，n，p
求 $(x+x^2+x^3+\ldots+x^n) \bmod p$ 的值。
$0<x,p<200000,1\leq n\leq 10^9$。

样例输入

```
3 1000000000 10007
```

样例输出

```
6215
```

参考代码

```
#include<cstdio>
#include<iostream>
#define LL long long
using namespace std;
int x,n,p;
```

```
int fast(LL a,int n,int p){
    LL s=1;
    while(n){
        if(n&1)s=(s*a)%p;
        a=(a*a)%p;
        n=n>>1;
    }
    return s;
}
LL sum(LL x,int n){//x+x^2+..+x^n
    if(n==1)return x%p;
    LL s=sum(x,n/2)%p;
    s=(s+s*fast(x,n/2,p))%p;
    if(n&1)s=(s+fast(x,n,p))%p;
    return s;
}
int main(){
    cin>>x>>n>>p;
    cout<<sum(x,n)<<endl;
    return 0;
}
```

3.求(a*b)%p

$1 \leq a, b, p \leq 10^18$

4.P3390【模板】矩阵快速幂

题目背景

矩阵快速幂

题目描述

给定 $n \times n$ 的矩阵 A，求 A^k。

输入格式

第一行两个整数 n, k
接下来 n 行，每行 n 个整数，第 i 行的第 j 的数表示 $A_{i,j}$。

输出格式

输出 A^k

共 n 行，每行 n 个数，第 i 行第 j 个数表示 $(A^k)_{i,j}$，每个元素对 $10^9 + 7$ 取模。

样例 #1

样例输入 #1

```
2 1
1 1
1 1
```

样例输出 #1

```
1 1
1 1
```

提示

【数据范围】
对于 100% 的数据：$1 \leq n \leq 100$，$0 \leq k \leq 10^{12}$，$|A_{i,j}| \leq 1000$

参考代码

```
#include<cstdio>
#include<iostream>
#include<cstring>
#define LL long long
#define MD 1000000007;
using namespace std;
struct Matrix{
    LL a[101][101];
};
Matrix A;
int n;
LL k;
Matrix operator *(const Matrix &A,const Matrix &B){
    Matrix C;
    memset(C.a,0,sizeof(C.a));
    for(int i=1;i<=n;i++)
        for(int j=1;j<=n;j++)
            for(int k=1;k<=n;k++)
                C.a[i][j]=(C.a[i][j]+A.a[i][k]*B.a[k][j])%MD;
    return C;
}
Matrix fast(Matrix A,LL k){
    Matrix S=A;
    k--;
    while(k){
        if(k&1) S=S*A;
        A=A*A;
        k=k>>1;
    }
    return S;
}
int main(){
    cin>>n>>k;
    for(int i=1;i<=n;i++)
        for(int j=1;j<=n;j++)cin>>A.a[i][j];
    Matrix ans=fast(A,k);
    for(int i=1;i<=n;i++){
        for(int j=1;j<n;j++)cout<<(ans.a[i][j])<<" ";
        cout<<(ans.a[i][n])<<endl;
    }
    return 0;
}
```

5.P1962斐波那契数列

题目描述

大家都知道，斐波那契数列是满足如下性质的一个数列：

$$F_n = \begin{cases} 1\ (n \leq 2) \\ F_{n-1} + F_{n-2}\ (n \geq 3) \end{cases}$$

请你求出 $F_n \bmod 10^9 + 7$ 的值。

输入格式

一行一个正整数 n

输出格式

输出一行一个整数表示答案。

样例 #1

样例输入 #1

```
5
```

样例输出 #1

```
5
```

样例 #2

样例输入 #2

```
10
```

样例输出 #2

```
55
```

提示

【数据范围】
对于 60% 的数据，$1 \leq n \leq 92$;
对于 100% 的数据，$1 \leq n < 2^{63}$。

参考代码

```
#include<cstdio>
#include<iostream>
#include<cstring>
#define LL long  long
```

```
#define MD 1000000007
using namespace std;
struct Matrix{
    long long a[3][3];
};
Matrix A;
LL n;
Matrix operator *(const Matrix &A,const Matrix &B){
    Matrix C;
    memset(C.a,0,sizeof(C.a));
    for(int i=1;i<=2;i++)
        for(int j=1;j<=2;j++)
            for(int k=1;k<=2;k++)
                C.a[i][j]=(C.a[i][j]+A.a[i][k]*B.a[k][j])%MD;
    return C;
}
Matrix fast(Matrix A,LL k){
    Matrix S=A;
    k--;
    while(k){
        if(k&1) S=S*A;
        A=A*A;
        k=k>>1;
    }
    return S;
}
int main(){
    cin>>n;
    A.a[1][1]=A.a[2][1]=A.a[1][2]=1;
    A.a[2][2]=0;
    if(n==0)cout<<0<<endl;
    else if(n==1||n==2)cout<<1<<endl;
    else{
        Matrix S=fast(A,n-2);
        cout<<(S.a[1][1]+S.a[2][1])%MD<<endl;
    }
    return 0;
}
```

6.P1939【模板】矩阵加速（数列）

题目描述

已知一个数列 a，它满足：

$$a_x = \begin{cases} 1 & x \in \{1,2,3\} \\ a_{x-1} + a_{x-3} & x \geq 4 \end{cases}$$

求 a 数列的第 n 项对 10^9+7 取余的值。

输入格式

第一行一个整数 T，表示询问个数。

以下 T 行，每行一个正整数 n。

输出格式

每行输出一个非负整数表示答案。

样例 #1

样例输入 #1

```
3
6
8
10
```

样例输出 #1

```
4
9
19
```

提示

- 对于 30% 的数据 $n \leq 100$；
- 对于 60% 的数据 $n \leq 2 \times 10^7$；
- 对于 100% 的数据 $1 \leq T \leq 100$，$1 \leq n \leq 2 \times 10^9$。

7.P4838 P哥破解密码

第二部分 ST表

1.P3865【模板】ST 表

题目背景

这是一道 ST 表经典题——静态区间最大值

请注意最大数据时限只有 0.8s，数据强度不低，请务必保证你的每次查询复杂度为 $O(1)$。若使用更高时间复杂度算法不保证能通过。

如果您认为您的代码时间复杂度正确但是 TLE，可以尝试使用快速读入：

```
inline int read()
{
    int x=0,f=1;char ch=getchar();
    while (ch<'0'||ch>'9'){if (ch=='-') f=-1;ch=getchar();}
    while (ch>='0'&&ch<='9'){x=x*10+ch-48;ch=getchar();}
    return x*f;
}
```

函数返回值为读入的第一个整数。

快速读入作用仅为加快读入，并非强制使用。

题目描述

给定一个长度为 N 的数列，和 M 次询问，求出每一次询问的区间内数字的最大值。

输入格式

第一行包含两个整数 N, M，分别表示数列的长度和询问的个数。

第二行包含 N 个整数（记为 a_i），依次表示数列的第 i 项。

接下来 M 行，每行包含两个整数 l_i, r_i，表示查询的区间为 $[l_i, r_i]$。

输出格式

输出包含 M 行，每行一个整数，依次表示每一次询问的结果。

样例 #1

样例输入 #1

```
8 8
9 3 1 7 5 6 0 8
1 6
1 5
2 7
2 6
1 8
4 8
3 7
1 8
```

样例输出 #1

```
9
9
7
7
9
8
7
9
```

提示

对于 30% 的数据，满足 $1 \leq N, M \leq 10$。

对于 70% 的数据，满足 $1 \leq N, M \leq 10^5$。

对于 100% 的数据，满足 $1 \leq N \leq 10^5$，$1 \leq M \leq 2 \times 10^6$，$a_i \in [0, 10^9]$，$1 \leq l_i \leq r_i \leq N$。

参考代码

```
#include<cstdio>
#include<iostream>
using namespace std;
```

```
const int N=1e5+10;
int f[N][21];
int a[N];
int Log[N];
int n,m;
int ask(int x,int y){
    int k=Log[y-x+1];
    return max(f[x][k],f[y-(1<<k)+1][k]);
}
int main(){
    scanf("%d%d",&n,&m);
    for(int i=1;i<=n;i++){
        scanf("%d",&a[i]);
        f[i][0]=a[i];
    }
    Log[1]=0;
    for(int i=2;i<=n;i++)Log[i]=Log[i>>1]+1;
    for(int j=1;j<=Log[n];j++)
        for(int i=1;i+(1<<(j-1))<=n;i++)
            f[i][j]=max(f[i][j-1],f[i+(1<<(j-1))][j-1]);
    for(int i=0;i<m;i++){
        int x,y;
        scanf("%d%d",&x,&y);
        printf("%d\n",ask(x,y));
    }
    return 0;
}
```

2.Frequent values （poj3368）

给定一个数组，其中的元素满足非递减顺序，要求对于一对起点和终点，回答出其中某个元素重复出现的最多次数。
比如对于-1 -1 1 1 1 1 3 10 10 10，若起点为1，终点为5，则重复出现最多的数是1，其次数为3。
输入：
n , q (1 ≤ n, q ≤ 100000);
a1 , … , an (-100000 ≤ ai ≤ 100000); ai ≤ ai+1;
以下q行：i and j (1 ≤ i ≤ j ≤ n)。
输出：
每个询问区间的最多次数。

Sample Input
10 3
-1 -1 1 1 1 1 3 10 10 10
2 3
1 10
5 10
Sample Output
1
4
3

参考代码

```
#include<cstdio>
```

```
#include<iostream>
#include<cmath>
using namespace std;
const int N=1e5+100;
int a[N],b[N];
int nxt[N];
int f[N][20];
int n,q,cnt,x,pre;
void st(){
    int k=log(n)/log(2);
    for(int i=1;i<=n;i++)f[i][0]=a[i];
    for(int j=1;j<=k;j++)
        for(int i=1;i<=n-(1<<j)+1;i++)
            f[i][j]=max(f[i][j-1],f[i+(1<<(j-1))][j-1]);
}
int ask(int l,int r){
    int k=log(r-l+1)/log(2);
    return max(f[l][k],f[r-(1<<k)+1][k]);
}

int main(){
    while(scanf("%d",&n)==1&&n>0){
        scanf("%d",&q);
        scanf("%d",&b[1]);
        a[1]=1;
        pre=b[1];
        for(int i=2;i<=n;i++){
            scanf("%d",&b[i]);
            if(b[i]==pre)a[i]=a[i-1]+1;
            else{
                pre=b[i];
                a[i]=1;
            }
        }
        nxt[n]=n;
        for(int i=n-1;i>=1;i--)
            if(b[i]==b[i+1])nxt[i]=nxt[i+1];
            else nxt[i]=i;
        st();
        for(int i=1;i<=q;i++){
            int l,r;
            scanf("%d%d",&l,&r);
            int x=nxt[l];
            if(r<=x)printf("%d\n",r-l+1);
            else{
                printf("%d\n",max(x-l+1,ask(x+1,r)));
            }
        }
    }
    return 0;
}
```

3.CF359D Pair of Numbers

题面翻译

题目描述

$Simon$ 有一个长度为 N 的正整数数列 $a_1, a_2, \cdots, a_n$ ， 现在他想找到这个数列中最长的一个区间，满足区间中有一个数 x 可以整除区间中任意数。

输入格式

第一行有一个正整数 N $(1 \leq N \leq 3 \times 10^5)$,表示数列的长度；
第二行有 N 个正整数 $a_1, a_2, \cdots, a_n$ $(1 \leq a_i \leq 10^6)$ ， 即为给出的数列。

输出格式

第一行输出两个正整数 cnt ， len ， 表示满足要求的最长区间的个数与长度。
第二行输出 cnt 个升序排列的正整数，表示所有满足要求的最长区间的左端点。
这里，区间的长度定义为**右端点减左端点**。

题目描述

Simon has an array $a_1, a_2, \ldots, a_n$, consisting of n positive integers. Today Simon asked you to find a pair of integers l, r $(1 <= l <= r <= n)$, such that the following conditions hold:

1. there is integer j ($l <= j <= r$), such that all integers $a_l, a_{l+1}, \ldots, a_r$ are divisible by a_j ;
2. value $r - l$ takes the maximum value among all pairs for which condition 1 is true;

Help Simon, find the required pair of numbers (l, r) . If there are multiple required pairs find all of them.

输入格式

The first line contains integer n ($1 <= n <= 3 \cdot 10^5$).

The second line contains n space-separated integers $a_1, a_2, \ldots, a_n$ $(1 <= a_i <= 10^6)$.

输出格式

Print two integers in the first line — the number of required pairs and the maximum value of $r - l$. On the following line print all l values from optimal pairs in increasing order.

样例 #1

样例输入 #1

```
5
4 6 9 3 6
```

样例输出 #1

```
1 3
2
```

样例 #2

样例输入 #2

```
5
1 3 5 7 9
```

样例输出 #2

```
1 4
1
```

样例 #3

样例输入 #3

```
5
2 3 5 7 11
```

样例输出 #3

```
5 0
1 2 3 4 5
```

提示

In the first sample the pair of numbers is right, as numbers $6, 9, 3$ are divisible by 3 .

In the second sample all numbers are divisible by number 1 .

In the third sample all numbers are prime, so conditions 1 and 2 are true only for pairs of numbers $(1, 1)$, $(2, 2)$, $(3, 3)$, $(4, 4)$, $(5, 5)$.

第三部分 LCA

1.P3379 【模板】最近公共祖先（LCA）

题目描述

如题，给定一棵有根多叉树，请求出指定两个点直接最近的公共祖先。

输入格式

第一行包含三个正整数 N, M, S，分别表示树的结点个数、询问的个数和树根结点的序号。

接下来 $N-1$ 行每行包含两个正整数 x, y，表示 x 结点和 y 结点之间有一条直接连接的边（数据保证可以构成树）。

接下来 M 行每行包含两个正整数 a, b，表示询问 a 结点和 b 结点的最近公共祖先。

输出格式

输出包含 M 行，每行包含一个正整数，依次为每一个询问的结果。

样例 #1

样例输入 #1

```
5 5 4
3 1
2 4
5 1
1 4
2 4
3 2
3 5
1 2
4 5
```

样例输出 #1

```
4
4
1
4
4
```

提示

对于 30% 的数据，$N \leq 10$，$M \leq 10$。

对于 70% 的数据，$N \leq 10000$，$M \leq 10000$。

对于 100% 的数据，$N \leq 500000$，$M \leq 500000$。

样例说明：

第一次询问：$2, 4$ 的最近公共祖先，故为 4。

第二次询问：$3, 2$ 的最近公共祖先，故为 4。

第三次询问：$3, 5$ 的最近公共祖先，故为 1。

第四次询问：$1, 2$ 的最近公共祖先，故为 4。

第五次询问：$4, 5$ 的最近公共祖先，故为 4。

故输出依次为 $4, 4, 1, 4, 4$。

2021/10/4 数据更新 @fstqwq：应要求加了两组数据卡掉了暴力跳。

参考代码

```
#include<cstdio>
#include<iostream>
#include<cmath>
using namespace std;
const int maxn=500000+100;
struct Edge{
int to,next;
```

```
};
Edge e[2*maxn];
int h[maxn];
int fa[maxn][20];
int d[maxn];
int n,m,s,tot,H;
void add(int u,int v){
    tot++;
    e[tot].to=v;
    e[tot].next=h[u];
    h[u]=tot;
}
void dfs(int u,int p){
    d[u]=d[p]+1;
    for(int i=h[u];i>0;i=e[i].next){
        int v=e[i].to;
        if(v==p)continue;
        fa[v][0]=u;
        dfs(v,u);
    }
}
int Lca(int x,int y){
    if(d[x]<d[y])swap(x,y);
    for(int i=H;i>=0;i--)
        if(d[fa[x][i]]>=d[y])x=fa[x][i];
    if(x==y)return x;
    for(int i=H;i>=0;i--)
        if(fa[x][i]!=fa[y][i])
            x=fa[x][i],y=fa[y][i];
    return fa[x][0];
}
int main(){
    scanf("%d%d%d",&n,&m,&s);
    tot=0;
    for(int i=1;i<n;i++){
        int u,v;
        scanf("%d%d",&u,&v);
        add(u,v);
        add(v,u);
    }
    H=log(n)/log(2)+1;
    d[0]=0;
    dfs(s,0);
    for(int j=1;j<=H;j++)
        for(int i=1;i<=n;i++)
            fa[i][j]=fa[fa[i][j-1]][j-1];
    for(int i=1;i<=m;i++){
        int u,v;
        scanf("%d%d",&u,&v);
        printf("%d\n",Lca(u,v));
    }
    return 0;
}
```

2.P4281 [AHOI2008]紧急集合 / 聚会

题目描述

欢乐岛上有个非常好玩的游戏，叫做“紧急集合”。在岛上分散有 n 个等待点，有 $n-1$ 条道路连接着它们，每一条道路都连接某两个等待点，且通过这些道路可以走遍所有的等待点，通过道路从一个点到另一个点要花费一个游戏币。

参加游戏的人三人一组，开始的时候，所有人员均任意分散在各个等待点上（每个点同时允许多个人等待），每个人均带有足够多的游戏币（用于支付使用道路的花费）、地图（标明等待点之间道路连接的情况）以及对话机（用于和同组的成员联系）。当集合号吹响后，每组成员之间迅速联系，了解到自己组所有成员所在的等待点后，迅速在 n 个等待点中确定一个集结点，组内所有成员将在该集合点集合，集合所用花费最少的组将是游戏的赢家。

小可可和他的朋友邀请你一起参加这个游戏，由你来选择集合点，聪明的你能够完成这个任务，帮助小可可赢得游戏吗？

输入格式

第一行两个正整数 n 和 m，分别表示等待点的个数（等待点也从 1 到 n 进行编号）和获奖所需要完成集合的次数。

随后 $n-1$ 行，每行两个正整数 a,b，表示编号为 a 和编号为 b 的等待点之间有一条路。

随后 m 行，每行用三个正整数 x,y,z，表示某次集合前小可可、小可可的朋友以及你所在等待点的编号。

输出格式

输出共 m 行，每行两个用空格隔开的整数 p,c。其中第 i 行表示第 i 次集合点选择在编号为 p 的等待点，集合总共的花费是 c 个游戏币。

样例 #1

样例输入 #1

```
6 4
1 2
2 3
2 4
4 5
5 6
4 5 6
6 3 1
2 4 4
6 6 6
```

样例输出 #1

```
5 2
2 5
4 1
6 0
```

提示

对于 40% 的数据，$n \leq 2 \times 10^3$，$m \leq 2 \times 10^3$。

对于 100% 的数据，$1 \leq x, y, z \leq n \leq 5 \times 10^5$，$1 \leq m \leq 5 \times 10^5$。

3.P2420让我们异或吧

题目描述

异或是一种神奇的运算,大部分人把它总结成不进位加法.

在生活中...xor运算也很常见。比如，对于一个问题的回答，是为1，否为0.那么：

（A是否是男生 ）xor （ B是否是男生） = A和B是否能够成为情侣

好了，现在我们来制造和处理一些复杂的情况。比如我们将给出一颗树，它很高兴自己有N个结点。树的每条边上有一个权值。我们要进行M次询问，对于每次询问，我们想知道某两点之间的路径上所有边权的异或值。

输入格式

输入文件第一行包含一个整数N，表示这颗开心的树拥有的结点数，以下有N-1行，描述这些边，每行有3个数，u,v,w,表示u和v之间有一条权值为w的边。接下来一行有一个整数M，表示询问数。之后的M行，每行两个数u,v，表示询问这两个点之间的路径上的权值异或值。

输出格式

输出M行，每行一个整数，表示异或值

样例 #1

样例输入 #1

```
5
1 4 9644
2 5 15004
3 1 14635
5 3 9684
3
2 4
5 4
1 1
```

样例输出 #1

```
975
14675
0
```

提示

对于40%的数据，有1 ≤ N，M ≤ 3000；

对于100%的数据，有1 ≤ N ，M≤ 100000。

参考代码

```
#include<cstdio>
#include<iostream>
#include<cmath>
using namespace std;
const int N=100010;
struct edge{
    int v,w,nxt;
}e[2*N];
int h[N];
int d[N];
char a[N];
int fa[N][20];
int g[N][20];
int n,m,tot=0,H;
inline void add(int u,int v,int w){
    e[++tot].v=v,e[tot].w=w,e[tot].nxt=h[u];
    h[u]=tot;
}
void dfs(int u,int p,int dep){
    d[u]=dep;
    for(int i=h[u];i>0;i=e[i].nxt){
        int v=e[i].v,w=e[i].w;
        if(v==p)continue;
        fa[v][0]=u;
        g[v][0]|=w;
        for(int j=1;j<=H;j++){
            fa[v][j]=fa[fa[v][j-1]][j-1];
            g[v][j]=g[v][j-1]^g[fa[v][j-1]][j-1];
        }
        dfs(v,u,dep+1);
    }
}
int lca(int x,int y){
    if(d[x]<d[y])swap(x,y);
    for(int i=H;i>=0;i--)
        if(d[fa[x][i]]>=d[y])x=fa[x][i];
    if(x==y)return y;
    for(int i=H;i>=0;i--)
        if(fa[x][i]!=fa[y][i])
            x=fa[x][i],y=fa[y][i];
    return fa[x][0];
}
int ask(int x,int p){
    int ans=0;
    for(int i=H;i>=0;i--)
        if(d[fa[x][i]]>=d[p]){
            ans=ans^g[x][i];
            x=fa[x][i];
        }
    return ans;
}
int main(){
    cin>>n;
    for(int i=1;i<n;i++){
        int u,v,w;
        cin>>u>>v>>w;
        add(u,v,w);
```

```
        add(v,u,w);
    }
    H=log(n)/log(2)+1;
    dfs(1,0,1);
    cin>>m;
    for(int i=0;i<m;i++){
        int u,v;
        cin>>u>>v;
        if(u==v) cout<<0<<endl;
        else{
            int p=lca(u,v);
            int ans=ask(u,p)^ask(v,p);
            cout<<ans<<endl;
        }
    }
    return 0;
}
```

4.P1967 [NOIP2013 提高组] 货车运输

题目描述

A 国有 n 座城市，编号从 1 到 n，城市之间有 m 条双向道路。每一条道路对车辆都有重量限制，简称限重。

现在有 q 辆货车在运输货物， 司机们想知道每辆车在不超过车辆限重的情况下，最多能运多重的货物。

输入格式

第一行有两个用一个空格隔开的整数 n, m，表示 A 国有 n 座城市和 m 条道路。

接下来 m 行每行三个整数 x, y, z，每两个整数之间用一个空格隔开，表示从 x 号城市到 y 号城市有一条限重为 z 的道路。
注意： $x \neq y$，两座城市之间可能有多条道路 。

接下来一行有一个整数 q，表示有 q 辆货车需要运货。

接下来 q 行，每行两个整数 x, y，之间用一个空格隔开，表示一辆货车需要从 x 城市运输货物到 y 城市，保证 $x \neq y$

输出格式

共有 q 行，每行一个整数，表示对于每一辆货车，它的最大载重是多少。
如果货车不能到达目的地，输出 -1。

样例 #1

样例输入 #1

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

样例输出 #1

```
3
-1
3
```

提示

对于 30% 的数据，$1 \leq n < 1000$，$1 \leq m < 10,000$，$1 \leq q < 1000$；

对于 60% 的数据，$1 \leq n < 1000$，$1 \leq m < 5 \times 10^4$，$1 \leq q < 1000$；

对于 100% 的数据，$1 \leq n < 10^4$，$1 \leq m < 5 \times 10^4$，$1 \leq q < 3 \times 10^4$，$0 \leq z \leq 10^5$。