小米 Online Judge 2019 年 01 月常规赛题解

本题解仅供大家参考，欢迎大佬们提供更为巧妙的方法~

灯

题解

题解一：

假设每个灯对于一个开关，并且这个开关控制除了当前灯外的n-1个灯。

分情况讨论：

- 当 n 为奇数，l 为偶数时：这种情况我们能把所有灯熄灭，所以不能全部变亮
- 当 n 为奇数，l 为奇数时：这种情况我们只需要将所有的亮的或者灭的全部按一遍即可
- 当 n 为偶数，l 为偶数时：这种情况我们所有的灭的灯全部按一遍即可
- 当 n 为偶数，l 为奇数时：这种情况我们所有的亮的灯全部按一遍即可

题解二：

如果避开一盏亮灯，改变其它 n-1 盏灯，那么将有 n-m+1 盏亮； 如果避开一盏暗灯，改变其它 n-1 盏灯，那么将有 n-m-1 盏亮； 也就是由 m 盏灯亮，通过一步可变为 n-m+1 盏亮，或 n-m-1 盏亮。

最后应该有n盏灯亮，那么倒数第二步就应该有1盏灯亮，倒数第三步就应该有n-2盏灯亮，..... 由后往前亮灯的盏数：n-0、1、n-2、3、n-4、5、n-6、.....，

结论就是：

- 如果 n 为偶数、m 为偶数：n-m
- 如果 n 为偶数、m 为奇数：m
- 如果 n 为奇数、m 为偶数：不可能
- 如果 n 为奇数、m 为奇数：n-m 和 m 两者之最小值

标程

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

int main(){
    long long n,l;
    while (cin >> n>>l) {
        if (n%2) {
            if (l%2==0) {
                puts("Impossible");
            } else {
                cout << min(l,n-l) << endl;
```

```
            }
        }
        else if(l%2) {
            cout << l << endl;
        } else {
            cout << n-l << endl;
        }
    }
}
```

> 本题由志愿者 WAFUN 提供

殊途同归

题解

题目大意：

在有铁轨直接相连的城市之间可以跑火车，在没有铁轨直接相连的城市之间可以修公路跑汽车。在两者不同时到达同一个城市的前提下，问汽车和火车从城市 1 到城市 n 走最优路的最大值是多少。

思路：

由于是没有铁轨直接相连的城市间可以修公路，所以两个城市之间要么有铁路，要么有公路。很明显，汽车和火车不会同时停靠在同一个城市。

所以，可以用 dijkstra 算法跑两次最短路，先对火车求最短路，然后对汽车求最短路，然后取最大值就可以了。

另外，m=0 时要输出 -1，否则会 RE。另外输出 -1 的情况就是火车到达不了 n 或者没有公路（有$\frac{n\cdot(n-1)}{2}$条边）的情况。

标程

```
#include<iostream>
#include<cstring>
#include<algorithm>

using namespace  std;
const int maxn=1100;
const int max1=1e7+1e6;
const int INF=0x3f3f3f3f;
int line[maxn][maxn],dis[max1];
bool vis[max1];
int Dijstra(int n,int x)
{
    memset(dis,0,sizeof(dis));
    for(int i=1;i<=n;i++)
    {
        dis[i]=line[x][i];
        vis[i]=false;
    }
```

```
    dis[1]=0;
    vis[x]=true;
    int Min,p;
    for(int i=1;i<=n;i++)
    {
        Min=INF;
        p=1;
        for(int j=1;j<=n;j++)
        {
            if(!vis[j]&&dis[j]<Min)
            {
                p=j;
                Min=dis[j];
            }
        }
        vis[p]=true;
        for(int j=1;j<=n;j++)
        {
            if(!vis[j]&&dis[p]+line[p][j]<dis[j])
            {
                dis[j]=dis[p]+line[p][j];
            }
        }
    }
}

int main()
{
    int n,m;
    while(cin>>n>>m)
    {
        memset(line,INF,sizeof(line));
        for(int i=1;i<=m;i++)
        {
            int a,b;
            cin>>a>>b;
           line[a][b]=line[b][a]=1;
        }
        Dijstra(n,1);
        int ans=dis[n];
        if(ans>=INF)
        {
            cout<<-1<<endl;
            break;
        }
        for(int i=1;i<=n;i++)
        {
            for(int j=1;j<=n;j++)
            {
            if(line[i][j]==1)line[i][j]=INF;
            else if(line[i][j]==INF)line[i][j]=1;
            }
        }
```

```
        Dijstra(n,1);
        ans=max(ans,dis[n]);
        if(ans>=INF)
        {
            cout<<-1<<endl;
        }
        else
            cout<<ans<<endl;
    }
}
```

> 本题由志愿者 云空 提供

星空

题解

题解一：

简化版题意： n 个点的图，每条边可以染成 k 种颜色中的一种，边数不限，不可有重边和自环，求非同构图的个数。

这个经典问题可以转化为 n 个点的完全图，每条边可以染成 $k+1$ 种颜色中的一种，求非同构的个数。然后把第 $k+1$ 种颜色的边视为不存在即可。

然后解法就同 [SHOI2006] 有色图 了，过程过于麻烦不再赘述，网上题解很多，我的该题题解可以参考 这里。

题解二：

对于这题，我们把轨道看作点，把能量桥看作边，那么这个模型就变成了一张图。

然后就变成了求n个点的k重图的个数。

对于k=1(简单图)，这道题等价于BZOJ 1488 图的同构

对于n重图, $ans=\sum(k+1)^{sum_{i1}}\times sum_{i2}$

sum_{i1} 为对于某一种置换i的循环个数，sum_{i2} 为该置换的个数。

因此用Polya定理对边的循环集进行优化后算出结果即可。

注意预处理逆元。

BZOJ 1488 题解

标程

标程一：

```
#include <cmath>
#include <cstdio>
```

```cpp
#include <cctype>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <algorithm>
#define N 61
#define mod 2333
#define gc getchar
#define rg register
using namespace std;
typedef long long ll;

inline int rd() {
  rg int x = 0;
  rg bool f = 0;
  rg char c = gc();
  while (!isdigit(c)) {
    if (c == '-') f = 1;
    c = gc();
  }
  while (isdigit(c)) {
    x = x * 10 + (c ^ 48);
    c = gc();
  }
  return f ? -x : x;
}

int n, m, ans, stk[N], nstk[N], cnt[N];

int fac[N] = {1, 1}, inv[N] = {1, 1}, ifac[N] = {1, 1};

inline int gcd(int x, int y) {
  rg int tmp;
  while (y) tmp = x, x = y, y = tmp % y;
  return x;
}

inline int qpow(int x, int t) {
  int res = 1;
  while (t) {
    if (t & 1) res = (1ll * res * x) % mod;
    x = (1ll * x * x) % mod; t >>= 1;
  }
  return res;
}

inline void getans(int top) {
  int F = 0, S = fac[n], ntop = 0;
  for (rg int i = 1, cntt; i <= top; ++i) {
    cntt = 1;
    nstk[++ntop] = stk[i];
    while (i < top && stk[i + 1] == stk[i]) ++cntt, ++i;
    cnt[ntop] = cntt;
```

```
  }
  for (rg int i = 1; i <= ntop; ++i) {
    F += nstk[i] / 2 * cnt[i];
    F += cnt[i] * (cnt[i] - 1) / 2 * nstk[i];
    for (rg int j = i + 1; j <= ntop; ++j)
      F += cnt[i] * cnt[j] * gcd(nstk[i], nstk[j]);
  }
  for (rg int i = 1; i <= ntop; ++i)
    S = (1ll * S * ifac[cnt[i]] % mod * qpow(inv[nstk[i]], cnt[i])) % mod;
  ans = (ans + 1ll * S * qpow(m, F)) % mod;
}

void dfs(const int &d, const int &rem, const int &nows) {
  if(!rem) getans(d);
  for (rg int i = nows; i <= rem; ++i) {
    stk[d + 1] = i;
    dfs(d + 1, rem - i, i);
  }
}

inline void work() {
  memset(cnt, 0, sizeof cnt);
  memset(stk, 0, sizeof stk);
  memset(nstk, 0, sizeof nstk);
  n = rd(); m = rd() + 1;
  ans = 0; dfs(0,n,1);
  printf("%lld\n", (1ll * ans * qpow(fac[n], mod-2)) % mod);
}

int main() {
  int t = rd();
  for (rg int i = 2; i < N; ++i) {
    fac[i] = (1ll * fac[i-1] * i) % mod;
    inv[i] = (1ll * (mod - mod / i) * inv[mod % i]) % mod;
    ifac[i] = (1ll * ifac[i-1] * inv[i]) % mod;
  }
  while(t--) work();
  return 0;
}
```

标程二：

```
#include <cstdio>
#include <cstring>
#include <iostream>
#include <algorithm>
#define N 2410
#define mod 2333
using namespace std;
int n,m,cnt,ans;
int powtwo[N],factor[N],invfac[N],num[N],val[N],gcd[N][N],inv[N];
int get_gcd(int x,int y)
```

```
{
    while(y)
    {
        int t=y;
        y=x%y;
        x=t;
    }
    return x;
}
int get_inv(int x,int y)
{
    int ret=1;
    while(y)
    {
        if(y&1)ret=(ret*x)%mod;
        x=(x*x)%mod;
        y>>=1;
    }
    return ret;
}
void init()
{
    powtwo[0]=factor[0]=invfac[0]=1;
    for(int i=1;i<=mod;i++)
    {
        factor[i]=factor[i-1]*i%mod;
    }
}
void dfs(int now_num,int left)
{
    if(left==0)
    {
        int retnow=0;
        int bot=1;
        for(int i=1;i<=cnt;i++)
        {
            retnow+=num[i]*(num[i]-1)/2*val[i]+val[i]/2*num[i];
            for(int j=i+1;j<=cnt;j++)
                retnow+=num[i]*num[j]*get_gcd(val[i],val[j]);
        }
        for(int i=1;i<=cnt;i++)
        {
            bot=(bot*get_inv(val[i],num[i])%mod*factor[num[i]])%mod;
        }
        bot=get_inv(bot,mod-2)*factor[n]%mod;
        ans=(ans+get_inv(m,retnow)*bot%mod)%mod;
    }
    if(now_num>left)return;
    dfs(now_num+1,left);
    for(int i=1;i*now_num<=left;i++)
    {
        val[++cnt]=now_num,num[cnt]=i;
        dfs(now_num+1,left-i*now_num);
```

```
            cnt--;
        }
    }
    int main()
    {
        int T;
        scanf("%d",&T);
        init();
        while(T--) {
        scanf("%d%d",&n,&m);
        ++m;
        ans = 0;
        dfs(1,n);
        ans=ans*get_inv(factor[n],mod-2)%mod;
        printf("%d\n",ans);
        }
    }
```

> 本题由志愿者 SGCollin 提供