数学补充

数学

错排公式

全错排公式
D(1)=0, D(2)=1
$D(n)=(n-1)(D(n-1)+D(n-2))$

n个里面m个错排：
$ans(n,m)=C_n^m D(m)$

Lucas定理

A、B是非负整数，p是质数。AB写成p进制：$A=a[n]a[n-1]...a[0]$，$B=b[n]b[n-1]...b[0]$。
则组合数$C(A,B)$与$C(a[n],b[n])*C(a[n-1],b[n-1])*...*C(a[0],b[0])modp$同余
即：$Lucas(n,m,p)=c(n\%p,m\%p)*Lucas(n/p,m/p,p)$

```
void init(long long p)
{
    factorial[0] = 1;
    for(int i = 1;i <= p;i++)
        factorial[i] = factorial[i-1]*i%p;
    //for(int i = 0;i < p;i++)
        //ni[i] = mod_pow(factorial[i],p-2,p);
}

long long Lucas(long long a,long long k,long long p) //求C(n,m)%p p最大为10^5。a,b可以很大！
{
    long long re = 1;
    while(a && k)
    {
        long long aa = a%p;long long bb = k%p;
        if(aa < bb) return 0; //这个是最后的改动！
        re = re*factorial[aa]*mod_pow(factorial[bb]*factorial[aa-bb]%p,p-2,p)%p;//这儿的求逆不可先处理
        a /= p;
        k /= p;
    }
    return re;
}
```

线性时间求阶乘逆元

```
void init() {
    fac[0] = fac[1] = 1;
    for(int i = 2; i <= N; i ++)
        fac[i] = fac[i-1] * i % mod;//预处理一下，阶乘
    inv[N]=mod_pow(fac[N], mod-2, mod); //Fac[N]^{MOD-2}
    for(int i = N - 1; i >= 0; i --) inv[i] = inv[i + 1] * (i + 1) % mod;
}
```

斯特林数

一.第二类Stirling数

定理：第二类Stirling数S(p,k)计数的是把p元素集合划分到k个不可区分的盒子里且没有空盒子的划分个数。
证明：元素在拿些盒子并不重要，唯一重要的是各个盒子里装的是什么，而不管哪个盒子装了什么。
递推公式有：

$$S(p,p)=1(p>=0)$$
$$S(p,0)=0(p>=1)$$
$$S(p,k)=k*S(p-1,k)+S(p-1,k-1)(1<=k<=p-1)$$

考虑将前p个正整数，1，2，.....p的集合作为要被划分的集合，把
{1,2,.....p}分到k个非空且不可区分的盒子的划分有两种情况：
(1)那些使得p自己单独在一个盒子的划分，存在有$S(p-1,k-1)$种划分个数
(2)那些使得p不单独自己在一个盒子的划分，存在有 $k*S(p-1,k)$种划分个数
考虑第二种情况，p不单独自己在一个盒子，也就是p和其他元素在一个集合里面，也就是说在没有放p之前，有p-1个元素已经分到了k个非空且不可区分的盒子里面（划分个数为$S(p-1,k$），那么现在问题是把p放在哪个盒子里面呢，有k种选择，所以存在有$k*S(p-1,k)$。

```
long long s[maxn][maxn];//存放要求的Stirling数
const long long mod=1e9+7;//取模

void init()//预处理
{
    memset(s,0,sizeof(s));
    s[1][1]=1;
    for(int i=2;i<=maxn-1;i++)
        for(int j=1;j<=i;j++)
    {
        s[i][j]=s[i-1][j-1]+j*s[i-1][j];
        if(s[i][j]>=mod)
            s[i][j]%=mod;
    }
}
```

二、第一类Stirling数

定理：第一类Stirling数$s(p,k)$计数的是把p个对象排成k个非空循环排列的方法数。
证明：把上述定理叙述中的循环排列叫做圆圈。递推公式为：
$s(p,p)=1(p>=0)$ 有p个人和p个圆圈，每个圆圈就只有一个人
$s(p,0)=0(p>=1)$ 如果至少有1个人，那么任何的安排都至少包含一个圆圈
$s(p,k)=(p-1)*s(p-1,k)+s(p-1,k-1)$
设人被标上$1,2,\ldots.p$。将这p个人排成k个圆圈有两种情况。第一种排法是在一个圆圈里只有标号为p的人自己，排法有$s(p-1,k-1)$个。第二种排法中，p至少和另一个人在一
个圆圈里。这些排法可以通过把1,2....p-1排成k个圆圈再把p放在1,2....p-1任何一人的左边得到，因此第二种类型的排法共有$(p-1)*s(p-1,k)$种排法。
在证明中我们所做的就是把{1,2,...,p}划分到k个非空且不可区分的盒子，然后将每个盒子中的元素排成一个循环排列。

```
long long s[maxn][maxn];//存放要求的第一类Stirling数
const long long mod=1e9+7;//取模

void init()//预处理
{
    memset(s,0,sizeof(s));
    s[1][1]=1;
    for(int i=2;i<=maxn-1;i++)
        for(int j=1;j<=i;j++)
    {
        s[i][j]=s[i-1][j-1]+(i-1)*s[i-1][j];
        if(s[i][j]>=mod)
            s[i][j]%=mod;
    }
```

```
}
```