Don't fake it till you made it

Fake it till you become it

Stirling数学习笔记

分类：

FNT DP (25) 多项式求逆

劼爷上的课现在才去整理…

第一类Stirling数

$s(n,m)$表示n个元素组成m个圆排列

由以上定义我们可以得出递推公式：

$s(n,m)=\sum_{i=0}^{n-1} s(n-i,m-1)*C_{n-1}^{i-1}*(i-1)!$

以及

$s(n,m)=s(n-1,m-1)+s(n-1,m)*(n-1)$

那么我们可以得到

$\prod_{i=1}^{i=n}(x+i-1)=\sum_{i=0}^{n} s(n,i)x^i$

然后可以我们就可以在$O(nlog_2n)$内求出一行Stirling数

假设我们已经有了

$F(x,n)=\prod_{i=1}^{i=n}(x+i-1)$

则

$F(x,2n)=\prod_{i=1}^{i=n}(x+i-1)*\prod_{i=1}^{i=n}(x+n+i-1)$

$=F(x,n)*G(x,n)$

其中

$[x^i]G(x,n)=\sum_{j=i}^{n} C_j^i*[x^j]F[x,n]*n^{j-i}$

然后就成了一个卷积形式

FNT即可

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


#define ll long long
const
    int Mod=998244353,p=3,D=Mod-1;

int Add(int a,int b)
{
    a+=b;
    if(a>=Mod)a-=Mod;
    if(a<0)a+=Mod;
    return a;
}

int Mul(int a,int b)
{
    return a*1ll*b%Mod;
}

int Pow(int a,int x)
```

关闭

```
{
    int res=1;
    for(;x;x>>=1,a=a*1ll*a%Mod)
        if(x&1)res=res*1ll*a%Mod;
    return res;
}
int Rev[400001];

void Rader(int n)
{
    for(int i=0;i<(1<<n);i++)Rev[i]=Rev[i>>1]>>1|((i&1)<<n-1);
}

int FNT(int *a,int n,int d)
{
    Rader(n);
    int len=1<<n;
    for(int i=0;i<len;i++)if(Rev[i]>i)swap(a[Rev[i]],a[i]);
    for(int i=1;i<len;i*=2)
    {
        int W=(d==1?Pow(p,D/2/i):Pow(p,D-D/2/i));
        for(int j=0;j<len;j+=2*i)
        {
            int W0=1;
            for(int k=0;k<i;k++)
            {
                int X=a[j+k],Y=Mul(W0,a[j+k+i]);
                a[j+k]=Add(X,Y);a[j+k+i]=Add(X,-Y);
                W0=Mul(W0,W);
            }
        }
    }
    int In=Pow(len,D-1);
    if(d==-1)for(int i=0;i<len;i++)a[i]=Mul(a[i],In);
}


void MUL(int*a,int alen,int *b,int blen,int *M,int &M1)
{
    int n=1;
    while((1<<n)<=alen+blen)n++;
    FNT(a,n,1);FNT(b,n,1);
    for(int i=0;i<(1<<n);i++)
        M[i]=Mul(a[i],b[i]);
    FNT(M,n,-1);
    M1=1<<n;
    while(M1&&!M[M1-1])M1--;
}
int Last[100001],Now[100001],L2,Last2[100001];
int F[100001],flen,G[100001],glen;
int Fact[100001],Inv[100001];
void Solve(int n)
{
    if(n==1){Last[0]=0,Last[1]=1;return;}
    int t=n>>1;
    Solve(t);
    flen=t;
    memset(F,0,sizeof(F));
    memset(G,0,sizeof(G));
    memset(Last2,0,sizeof(Last2));
    for(int i=0;i<=t;i++)
        F[t-i]=Mul(Last[i],Fact[i]),Last2[i]=Last[i];
    G[0]=1;
    for(int i=1;i<=t;i++)
        G[i]=Mul(Mul(G[i-1],Fact[i-1]),Mul(Inv[i],t));
    MUL(F,t+1,G,t+1,Now,L2);
    L2--;
    for(int i=0;i<=t;i++)
        Last[t-i]=Mul(Now[i],Inv[t-i]);
    if(n&1)
        {
```

关闭

```
            for(int i=t+1;i;i--)
                Last[i]=Add(Last[i-1],Mul(n-1,Last[i]));
            Last[0]=Mul(n-1,Last[0]);
        }
    MUL(Last,t+2,Last2,n-t+1,Last,L2);
}


int main()
{
    int n;
    scanf("%d",&n);
    Fact[0]=1;
    for(int i=1;i<=n;i++)Fact[i]=Mul(Fact[i-1],i);
    Inv[n]=Pow(Fact[n],Mod-2);
    for(int i=n-1;~i;i--)
    Inv[i]=Mul(Inv[i+1],i+1);
    Solve(n);
    int l,r;
    for(int i=0;i<=n;i++)
     printf("%d  ",Last[i]);
    return 0;
}
```

第二类$Stirling$数：

$S(n,m)$表示n个元素分成m个集合的方案数

那么由定义可得递推式:

$$S(n,m)=S(n,m-1)+S(n-1,m)*m$$

并且

$$k^n=\sum_{m=0}^{k}A_k^m*S(n,m)=\sum_{m=0}^{k}m!*C_k^m*S(n,m)$$

可以用n个球放入k个篮子,允许存在空篮子的方案数

由上式 我们经过二项式反演之后可以得到

$$S(n,m)=\sum_{k=0}(-1)^{m-k}*C_m^k*(m-k)^n$$

$$S(n,m)=\sum_{k=0}(-1)^k*\frac{1}{k!}*\frac{1}{m-k!}*(m-k)^n$$

同样可以经过FFT $(nlog_2n)$得出

```
#include<cstdio>
#include<iostream>
#include<cstring>
#include<cstdlib>
#include<cmath>
#include<algorithm>
using namespace std;

#define ll long long
const
    int Mod=998244353,p=3,D=Mod-1;

int Add(int a,int b)
{
    a+=b;
    if(a>=Mod)a-=Mod;
    if(a<0)a+=Mod;
    return a;
}

int Mul(int a,int b)
{
    return a*1ll*b%Mod;
}

int Pow(int a,int x)
{
    int res=1;
    for(;x;x>>=1,a=a*1ll*a%Mod)
        if(x&1)res=res*1ll*a%Mod;
    return res;
```

关闭

```
}

int Rev[100001];
void Rader(int n)
{
    for(int i=0;i<(1<<n);i++)Rev[i]=(Rev[i>>1]>>1)|((i&1)<<n-1);
}

void FNT(int *a,int n,int d)
{
    Rader(n);
    int len=1<<n,Inv=Pow(len,Mod-2);
    for(int i=0;i<len;i++)if(Rev[i]>i)swap(a[Rev[i]],a[i]);
    for(int i=1;i<len;i*=2)
    {
        int W0=(d==1?(Pow(p,D/2/i)):(Pow(p,D-D/2/i)));
        for(int j=0;j<len;j+=2*i)
            {
                int W=1;
                for(int k=0;k<i;k++)
                {
                    int X=a[k+j],Y=Mul(a[k+j+i],W);
                    a[k+j]=Add(X,Y);
                    a[k+i+j]=Add(X,-Y);
                    W=Mul(W,W0);
                }
            }
    }
    if(d==-1)
    for(int i=0;i<len;i++)a[i]=Mul(a[i],Inv);

}

int F[100001],G[100001],Fact[100001],Inv[100001];
int Ans[100001];
int main()
{
    int n;
    scanf("%d",&n);
    int B=1;
    while((1<<B)<=n)B++;B++;
    Fact[0]=1;
    for(int i=1;i<=n;i++)Fact[i]=Mul(Fact[i-1],i);
    Inv[n]=Pow(Fact[n],Mod-2);
    for(int i=n-1;~i;i--)Inv[i]=Mul(Inv[i+1],i+1);
    for(int i=0;i<=n;i++)
    {
        F[i]=Add(0,Mul((i&1?-1:1),Inv[i]));
        G[i]=Mul(Pow(i,n),Inv[i]);
    }
    FNT(F,B,1);
    FNT(G,B,1);
    for(int i=0;i<(1<<B);i++)
    Ans[i]=Mul(F[i],G[i]);
    FNT(Ans,B,-1);
    for(int i=0;i<=n;i++)
    printf("%d ",Ans[i]);
    return 0;
}
```


第二类Stirling数还有一个好性质:

$x^n=\sum_{i=0}^{n}A_x^i*S(n,i)$

依旧考虑x个盒子放n个不同的球,可以存在空盒子

则左式可以看做枚举非空盒子个数 然后按顺序取出i个盒子放入球

根据这个性质我们可以很方便的处理一类代价为n次方的问题

HDU4625 JZPTREE

给出一棵n个节点边权为1的树，对每一个节点i，输出
$\sum_{j=1}^{n} dist(i,j)^m$
$n \le 5 \times 10^4, m \le 500$
注意到此题m较小,可以用上述方法求解
唯一的问题就是如何从A_x^i得出A_{x+1}^i
对于A_{x+1}^i我们有递推公式
$A_{x+1}^i = A_x^i + i * A_x^{i-1}$
然后树形DP处理出每一个点 u 的A_x^i和,最后乘上相应的Stirling数即可得到答案
时间复杂度:$O(n*m)$

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

char c;
inline void read(int&a)
{
    a=0;do c=getchar();while(c<'0'||c>'9');
    while(c<='9'&&c>='0')a=(a<<3)+(a<<1)+c-'0',c=getchar();
}
const
    int Mod=10007;
int S[501][603];
int Low[60001][603];
int High[60001][603];
int n,m;
struct Chain
{
    Chain*next;
    int u;
}*Head[50002];
Chain R[100001];
int A;
inline void Add(int a,int b)
{Chain*tp=R+A;A++;tp->next=Head[a];Head[a]=tp;tp->u=b;}
int F[60001];

inline int Up(int u,int i)
{

    return ((i>=1?i*1ll*(Low[u][i-1]):0)+(i>=0?Low[u][i]:0))%Mod;
}

inline int add(int a,int b)
{

    a+=b;
    if(a>=Mod)a-=Mod;
    if(a<0)a+=Mod;
    return a;
}

inline int mul(int a,int b)
{

    a=a*b;
    a%=Mod;
    return a;
}
void DFS(int u,int f)
{
    F[u]=f;

    for(Chain*tp=Head[u];tp;tp=tp->next)
    if(tp->u!=f)
        {

```

关闭

```
            DFS(tp->u,u);
            for(int i=0;i<=m;i++)
                Low[u][i+1]=add(Low[u][i+1],Up(tp->u,i+1));
            Low[u][0]=add(Low[u][0],Low[tp->u][0]);

        }
    Low[u][0]=add(1,Low[u][0]);
}

void DFS2(int u,int f)
{
    for(Chain*tp=Head[u];tp;tp=tp->next)
    if(tp->u!=f)
        {
            for(int i=0;i<=m;i++)
                High[tp->u][i]
                =add(add(add(High[u][i],-Up(tp->u,i)),mul(i,i>=1?add(High[u][i-1],-Up(tp->
            DFS2(tp->u,u);
        }

}



int Calc(int u)
{
    int res=0,f=F[u];
    if(u==1)
    {
        for(int i=1;i<=m;i++)
        {
            res=add(res,mul(S[m][i],Low[u][i]));
        }
        return res;
    }
    for(int i=1;i<=m;i++)
    {
        res=
        add(res,
            mul(S[m][i],
                High[u][i]
            ));
    }
    return res;
}



int main()
{
freopen("self.in","r",stdin);
freopen("self.out","w",stdout);
    S[0][0]=1;
    for(int i=1;i<=500;i++)
        for(int j=1;j<=500;j++)
            S[i][j]=(S[i-1][j-1]+S[i-1][j]*j)%Mod;
    int T;
    read(T);
    while(T--)
    {
        A=0;
        memset(F,0,sizeof(F));
        memset(Low,0,sizeof(Low));
        memset(High,0,sizeof(High));
        memset(Head,0,sizeof(Head));
        read(n),read(m);
        for(int i=1;i<n;i++)
        {
            int a,b;
            read(a),read(b);
            Add(a,b),Add(b,a);
```

关闭

```
        }
        DFS(1,1);
        for(int i=0;i<=m;i++)High[1][i]=Low[1][i];
        DFS2(1,1);
        for(int i=1;i<=n;i++)
            printf("%d\n",Calc(i));

    }
}
```

Hackerrank Costly Graphs

题目大意:

求所有节点为n的图的权值和

这里图G的权值和定义为$\sum_{u\in G} D(u)^m$

$D(u)$为节点u的度

m为给定常数

$1 \le n \le 10^9$

$1 \le m \le 2 \cdot 10^5$

显然每个节点对于答案贡献独立且相等 我们考虑其中任意节点u的贡献

则有贡献

$$W(u) = 2^{C_{n-1}^2} * \sum_{i=0}^{n-1} i^m * C_{n-1}^i$$
$$W(u) = 2^{C_{n-1}^2} * \sum_{i=0}^{n-1} C_{n-1}^i * \sum_{t=0}^{m} A_i^t * S(m,t)$$
$$W(u) = 2^{C_{n-1}^2} * \sum_{i=0}^{n-1} \frac{(n-1)!}{(n-1-i)!} * \sum_{t=0}^{min(m,i)} \frac{1}{(i-t)!} * S(m,t)$$
$$W(u) = (n-1)! * 2^{C_{n-1}^2} * \sum_{t=0}^{m} S(m,t) * \sum_{i=t}^{n-1} \frac{1}{(n-1-i)!*(i-t)!}$$
$$W(u) = (n-1)! * 2^{C_{n-1}^2} * \sum_{t=0}^{m} S(m,t) * \sum_{i=0}^{n-1-t} \frac{1}{(n-1-t-i)!*(i)!}$$
$$W(u) = (n-1)! * 2^{C_{n-1}^2} * \sum_{t=0}^{m} S(m,t) * \frac{1}{(n-1-t)!} * 2^{n-1-t}$$
$$W(u) = 2^{C_{n-1}^2} * \sum_{t=0}^{m} S(m,t) * A_{n-1}^t * 2^{n-1-t}$$

答案就是$n * W(u)$

注意幂次的取模需要费马小定理..

```
#include<cstdio>
#include<iostream>
#include<cstring>
#include<cmath>
#include<algorithm>
#include<cstdlib>
using namespace std;

const
    int Mod=1005060097,p=7;

char c;
inline void read(int&a){a=0;do c=getchar();while(c<'0'||c>'9');while(c<='9'&&c>='0')a=(a<<

int Pow(int a,int x)
{
    int res=1;
    for(;x;x>>=1,a=a*1ll*a%Mod)
    if(x&1)res=res*1ll*a%Mod;
    return res;
}
int mul(int a,int b){return a*1ll*b%Mod;}
int mul(int a,int b,int Mod){return a*1ll*b%Mod;}
int add(int a,int b)
{
    a=a+b;
    if(a>=Mod)a-=Mod;
    if(a<0)a+=Mod;
    return a;
}

int Rev[1000001];
```

关闭

```
inline void Rader(int n)
{
    for(int i=0;i<(1<<n);i++)Rev[i]=(Rev[i>>1]>>1)|((i&1)<<n-1);
}

void FNT(int *a,int n,int l)
{
    int len=1<<n,I=Pow(len,Mod-2);
    Rader(n);
    for(int i=0;i<len;i++)
        if(Rev[i]>i)swap(a[Rev[i]],a[i]);
    for(int i=1;i<len;i*=2)
    {
        int W=Pow(p,l==1?(Mod-1)/2/i:(Mod-1-(Mod-1)/2/i));
        for(int j=0;j<len;j+=2*i)
        {
            int W0=1;
            for(int k=0;k<i;k++)
            {
                int x=a[j+k],y=mul(a[j+k+i],W0);
                a[j+k]=add(x,y);
                a[j+k+i]=add(x,-y);
                W0=mul(W0,W);
            }
        }
    }
    if(l==-1)
        for(int i=0;i<len;i++)a[i]=mul(a[i],I);
}

int Fact[1000001],Inv[1000001];
int F[1000001],G[1000001],S[1000001];

int C2(int n)
{
    if(n&1)return mul(n,n-1>>1,Mod-1);
        return mul(n>>1,n-1,Mod-1);

}

int main()
{
    int T;
    Fact[0]=1;
    for(int i=1;i<=200000;i++)Fact[i]=mul(Fact[i-1],i);
    Inv[200000]=Pow(Fact[200000],Mod-2);
    for(int i=199999;~i;i--)Inv[i]=mul(Inv[i+1],i+1);
    read(T);
    while(T--)
    {
        int n,m;
        read(n),read(m);
        for(int i=0;i<=m;i++)
            F[i]=add(0,(i&1?-1:1)*Inv[i]);
        for(int i=0;i<=m;i++)
            G[i]=mul(Pow(i,m),Inv[i]);
        int B=1;
        while((1<<B)<=m+m+1)B++;
        FNT(F,B,1);FNT(G,B,1);
        for(int i=0;i<(1<<B);i++)
        S[i]=mul(F[i],G[i]);
        FNT(S,B,-1);
        int ans=0,A=1,To=Pow(2,n-1),I=(Mod>>1)+1;
        for(int i=0;i<=m;To=mul(To,I),A=mul(A,(n-1-(i++))))
        ans=add(ans,mul(A,mul(To,S[i])));
        ans=mul(ans,mul(Pow(2,C2(n-1)),n));
        printf("%d\n",ans);
        for(int i=0;i<(1<<B);i++)F[i]=G[i]=S[i]=0;
    }
    return 0;
}
```

关闭

同时Stirling数可以应用于幂级数..

再来看看小火车在今年NOI十连测上的一道题:

一张无向图的权值为联通块个数的m次方，问所有n个点的带标号的无向图的权值和。
答案对998244353取模。
$T \le 1000, n \le 30000, m \le 15$。
构造一个多项式$L = \sum_{i=1}^{n} x_i$
其中x_i表示连通块i是否存在
则L^m为该图的权值
由扩展二项式定理可得
项 $\prod x_{a_i}^{t_i}$ 的系数为 $\frac{m!}{\prod t_i!}$，其中 $\sum ti = m$
考虑所有由 x_{a_1} 至 x_{a_k} 组成的项的系数和
对于任意$\sum_{i=1}^{k} ti = m$
我们有$\sum \frac{m!}{\prod t_i!} = \sum S(m,k) * k!$

则对于某 k 个联通块，如果它们在图中同时出现，那么贡献为
$S(m,k) \times k!$
设f_i为i个点的连通图数目
$f_i = 2^{C_i^2} - \sum_{j=1}^{i-1} C_{i-1}^{j-1} * f_j * 2^{C_{i-j}^2}$
我们可以使用多项式求逆的方法求出f
设$g_{i,j}$为i个点j个块的无向图个数
有$g_{i,j} = \sum_{k=1}^{i} C_i^k * f_k * g_{i-k,j-1}$
可以用卷积算出
答案即为$\sum_{j=1}^{m} j! * S(m,j) \sum_{i=1}^{n} C_n^i * g_{i,j} * 2^{C_{n-i}^2}$
可以把$\sum_{i=1}^{n} C_n^i * g_{i,j} * 2^{C_{n-i}^2}$看成$T(j,n)$
m次卷积预处理之后$O(m)$累加即可
时间复杂度$O(Tm + mnlog_2n)$

多项式求逆的话具体看2015年集训队鏼爷论文
自己的代码常数巨大

```
#include<cstdio>
#include<iostream>
#include<cmath>
#include<cstring>
using namespace std;
const
    int Mod=998244353,p=3,D=Mod-1;


int Pow(int a,int x)
{
    int res=1;
    for(;x;x>>=1,a=a*1ll*a%Mod)
        if(x&1)res=res*1ll*a%Mod;
    return res;
}
inline
int Mul(int a,int b)
{return a*1ll*b%Mod;}
inline int Add(int a,int b)
{
    a+=b;
    if(a>=Mod)a-=Mod;
    if(a<0)a+=Mod;
    return a;
}

int Rev[2000001];
int Rader(int n)
```

```
{
    for(int i=0;i<1<<n;i++)Rev[i]=(Rev[i>>1]>>1)|((i&1)<<n-1);
}

int FNT(int *a,int n,int f)
{
    int len=1<<n,I=Pow(len,D-1);
    Rader(n);
    for(int i=0;i<len;i++)if(Rev[i]>i)swap(a[Rev[i]],a[i]);
    for(int i=1;i<len;i*=2)
    {
        int W=Pow(p,f==1?D/2/i:(D-D/2/i));
        for(int j=0;j<len;j+=i*2)
        {
            int W0=1;
            for(int k=0;k<i;k++)
            {
                int x=a[j+k],y=Mul(W0,a[i+j+k]);
                a[j+k]=Add(x,y),a[i+j+k]=Add(x,-y);
                W0=Mul(W0,W);
            }
        }
    }
    if(f==-1)
    for(int i=0;i<len;i++)a[i]=Mul(a[i],I);
}

int T[200001],A[200001];
int Fact[100001],Inv[100001];

int C(int x)
{
    return (x*1ll*(x-1)>>1)%D;
}

int B[200001],Ca[200001];

void Find(int Len)
{
    if(Len==1)
        {B[0]=1;return;}
    int t=1;
    Find(Len>>1);
    memset(Ca,0,sizeof(Ca));
    memset(A,0,sizeof(A));
    while((1<<t)<=3*Len)t++;
    for(int i=0;i<Len;i++)
    A[i]=T[i];
    FNT(A,t,1);
    FNT(B,t,1);
    for(int i=0;i<1<<t;i++)
        Ca[i]=Add(Mul(2,B[i]),-Mul(B[i],Mul(B[i],A[i])));
    FNT(A,t,-1);
    FNT(Ca,t,-1);
    FNT(B,t,-1);
    memset(B,0,sizeof(B));
    for(int i=0;i<Len;i++)
        B[i]=Ca[i];

}
int S[101][101];

int G[2][100001];

int F[100001];
int ANS[101][100001];
int BIT[100001];
int main()
{
    freopen("self.in","r",stdin);
    freopen("self.out","w",stdout);
```

```

    int N=16384*2-1;
    Fact[0]=1;
    int P=116195171;
    for(int i=1;i<=N;i++)Fact[i]=Mul(i,Fact[i-1]);
    Inv[N]=Pow(Fact[N],D-1);
    for(int i=N-1;~i;i--)Inv[i]=Mul(i+1,Inv[i+1]);
    for(int i=1;i<=N;i++)
        T[i]=Mul(Pow(2,C(i)),Inv[i]);
    T[0]=1;
    Find(N+1);
    int TTT=16;
    FNT(B,TTT,1);
    memset(T,0,sizeof(T));
    for(int i=1;i<=N;i++)
        T[i]=Mul(Inv[i-1],Pow(2,C(i)));
    FNT(T,TTT,1);
    for(int i=0;i<1<<TTT;i++)
        F[i]=Mul(T[i],B[i]);
    FNT(F,TTT,-1);
    B[0]=0;
    int cas;
    S[1][1]=1;
    for(int i=2;i<=100;i++)
    for(int j=1;j<=100;j++)
    S[i][j]=Add(S[i-1][j-1],Mul(j,S[i-1][j]));


    N=1;
    int n=16384*2,m=15,now=1,next=0;
    G[now][1]=1;
    while((1<<N)<n)N++;
    N++;

    for(int i=1;i<=n;i++)
        G[now][i]=Mul(F[i],Fact[i-1]);
    memset(T,0,sizeof(T));
    for(int i=1;i<=n;i++)
        T[i]=F[i];
    T[0]=0;
    FNT(T,N,1);
    for(int i=1;i<n;i++)
        BIT[i]=Mul(Inv[i],Pow(2,C(i)));
    BIT[0]=1;
    FNT(BIT,N,1);
    for(int j=1;j<=m;j++,next^=1,now^=1)
    {
        for(int i=1;i<=n;i++)
            G[now][i]=Mul(G[now][i],Inv[i]);
        FNT(G[now],N,1);
        for(int i=0;i<1<<N;i++)
            ANS[j][i]=Mul(BIT[i],G[now][i]);
        FNT(ANS[j],N,-1);
        memset(G[next],0,sizeof(G[next]));
        for(int i=0;i<1<<N;i++)
            G[next][i]=Mul(G[now][i],T[i]);
        FNT(G[next],N,-1);
        for(int i=n+1;i<1<<N;i++)
            G[next][i]=0;
        for(int i=1;i<=n;i++)
            G[next][i]=Mul(G[next][i],Fact[i-1]);
        G[next][0]=0;
    }


    scanf("%d",&cas);
    while(cas--)
    {
        int ans=0;
        int n,m;
        memset(G,0,sizeof(G));scanf("%d%d",&n,&m);
```

```
        for(int j=1;j<=m;j++)
            ans=Add(ans,Mul(S[m][j],Mul(Fact[n],Mul(Fact[j],ANS[j][n]))));
        printf("%d\n",ans);
    }
    return 0;
}
```

还有利用反演的:

$f_i = \sum_{j=0}^{i} S(i,j) g_i$

为

$g_i = \sum_{j=0}^{i} (-1)^{i-j} s(i,j) f_j$

的充要条件

~~然后容斥的这里就埋个坑...什么时候会做了再说吧~~

关于容斥:

满足任意两行或者任意两列都不相同的 n × m 的数字矩阵有多少个，

每一个格子内的数必须是 [1, C] 内的整数。对一个大质数(10^9+7)取模。

$n, m, C \le 4000$

首先考虑如何容斥..

f_i表示列上最多有i个等价类每行不重复的方案数

g_i表示列上刚好有i个等价类每行不重复的方案数

则有$f_i = S(m,i) * A_{m^i}^{n} = \sum_{i=1}^{i} \frac{g_j}{S(m,j)} * S(m,i)$

然后就可以以$O(m^2)$暴力啦

```
#include<cstdio>
#include<cstring>
#include<cstdlib>
#include<cmath>

using namespace std;
const
    int Mod=1000000007;
#define ll long long
inline int Mul(int a,int b)
{
    return a*1ll*b%Mod;
}

inline int Add(int a,int b)
{
    a+=b;
    if(a>=Mod)a-=Mod;
    if(a<0)a+=Mod;
    return a;
}

inline int Pow(int a,int x)
{
    int res=1;
    for(;x;x>>=1,a=a*1ll*a%Mod)
    if(x&1)res=res*1ll*a%Mod;
    return res;
}

int F[10001],G[10001];
int S[4001][4001];
class CountTables{
    public:
int howMany(int n,int m,int C)
{
    S[1][1]=1;
    for(int i=2;i<=4000;i++)
    for(int j=1;j<=4000;j++)S[i][j]=Add(S[i-1][j-1],Mul(j,S[i-1][j]));
    int INV=1;
    INV=Pow(INV,Mod-2);
```

```
    int D=C,P=1;
    for(int i=1;i<=m;i++)
        {
            int V=1;
            for(int j=0;j<n;j++)
                V=Mul(V,Add(D,-j));
            F[i]=V;
            D=Mul(D,C);
            P=Mul(P,i);
        }
    for(int i=1;i<=m;i++)
        {
            G[i]=F[i];
            for(int j=1;j<i;j++)
                G[i]=Add(G[i],-Mul(G[j],S[i][j]));
        }
    return Mul(G[m],1);
    return Mul(G[m],Pow(INV,Mod-2));
}}Gh;


int main()
{
    int n,m,C;
    scanf("%d%d%d",&n,&m,&C);
    printf("%d\n",Gh.howMany(n,m,C));
}
```

To Be continued

上一篇 Hackerrank XOR Subsequences

下一篇 HDU5766 Filling