Wannafly挑战赛18 A,B,C

[比赛链接]

这场比赛我第二题被卡了很久~~当然不能说是大佬的错~~。

最终91名，$T2$交了太多次，罚时极高QwQ。

A：序列

题目大意：让你生成一个序列，满足$a_1=1,a_n=1$，且$a_i=a_{i-1}||a_i=-2*a_{i-1}||a_i=\frac{1}{2}a_{i-1}(1<i<n)$求方案总数。

·组合数学

显然，将序列可以表示成：$1*x_1*x_2*...x_{n-1}=1(x_i=1||-2||\frac{1}{2})$。

然后因为我们最后要得到的数为1，所以-2和0.5的数量必定一样，且这两个一样的数必是偶数。

所以可得$Ans=\sum_{i=0,i\in 2*k,k\in Z}^{n/2}C(n,i)*C(n-i,i)$

```
#include <cstdio>
using namespace std;
long long N,Mod=1e9+7,fac[1005],Ans;
long long inv(long long x,long long y){
    long long res=1;while(y){
        if(y&1)res=res*x%Mod;x=x*x%Mod,y>>=1;
    }return res;
}
long long C(long long x,long long y){
    return fac[x]*inv(fac[y],Mod-2)%Mod*inv(fac[x-y],Mod-2)%Mod;
}
int main()
{
    scanf("%lld",&N);
    fac[0]=fac[1]=1;
    for(int i=2;i<=N;i++)fac[i]=fac[i-1]*i%Mod;N--;
    for(int i=0;i<<1<=N;i+=2)(Ans+=C(N,i)*C(N-i,i)%Mod)%=Mod;
    printf("%lld",Ans);
    return 0;
}
```

B：随机数

题目大意：给定a,b，你每次有$\frac{a}{10000}$的概率生成1，$1-\frac{a}{10000}$的概率生成0，求运行n次后，1的个数为奇数的概率，对1e9+7取膜。

·矩阵加速DP转移。

设$f_{i,0}$表示前i次生成1数量为偶数的概率，$f_{i,1}$表示前i次生成1数量为奇数的概率。

设$p=\frac{a}{10000},q=1-\frac{a}{10000}$

可得转移方程：

> $f_{i,0}=p*f_{i,1}+q*f_{i,0}$
>
> $f_{i,1}=p*f_{i,0}+q*f_{i,1}$

对于这个转移我们不难发现可以表示成矩阵的形式：$\begin{bmatrix}1&0\\0&1\end{bmatrix}*\begin{bmatrix}q&p\\p&q\end{bmatrix}^n$

因为n的范围很大，这样的话我们就一位一位处理。

对于每一位的处理：$Ans=Ans^{10}*f^{c_i}$，Ans为$\begin{bmatrix}1&0\\0&1\end{bmatrix}$，$f$为$\begin{bmatrix}q&p\\p&q\end{bmatrix}$

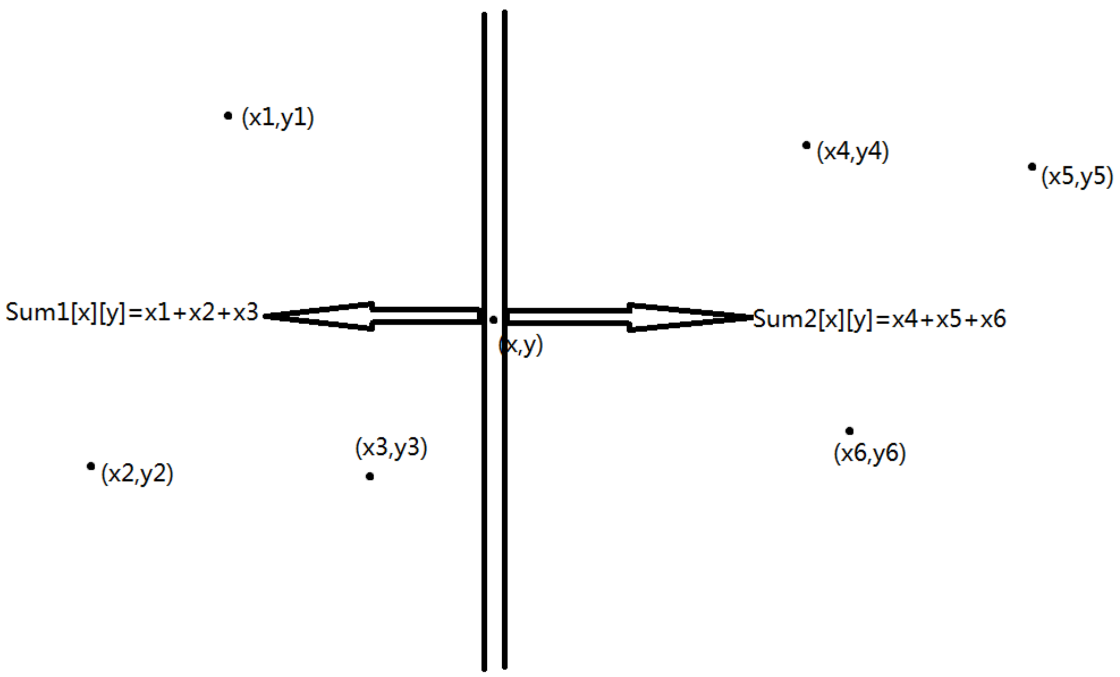
```
#include <cstdio>
#include <cstring>
using namespace std;
char q[1000005];
int N,Mod=1e9+7,j,k;
int c[1000005];
struct martix{int a[2][2];}f,Ans,z,p,tmp1,tmp2;
inline martix operator *(const martix &x,const martix &y){
    z.a[0][0]=z.a[0][1]=z.a[1][0]=z.a[1][1]=0;
    z.a[0][0]=(1ll*x.a[0][0]*y.a[0][0]+1ll*x.a[0][1]*y.a[1][0])%Mod;
    z.a[0][1]=(1ll*x.a[0][0]*y.a[0][1]+1ll*x.a[0][1]*y.a[1][1])%Mod;
    z.a[1][0]=(1ll*x.a[1][0]*y.a[0][0]+1ll*x.a[1][1]*y.a[1][0])%Mod;
    z.a[1][1]=(1ll*x.a[1][0]*y.a[0][1]+1ll*x.a[1][1]*y.a[1][1])%Mod;
    return z;
}
inline martix operator *(martix x,int y){
    p.a[0][0]=p.a[1][1]=1,p.a[0][1]=p.a[1][0]=0;
    while(y){if(y&1)p=p*x;x=x*x;y>>=1;}return p;
}
inline int inv(int x,int y){
    int res=1;while(y){
        if(y&1)res=1ll*res*x%Mod;x=1ll*x*x%Mod,y>>=1;
    }return res;
}
int main()
{
    scanf("%d",&N),scanf("%s",q);
    //if(N==0) return puts("0"),0;
    N=1ll*N*inv(10000,Mod-2)%Mod;
    register int len=strlen(q),i;
    f.a[0][1]=N,f.a[0][0]=(1-N+Mod)%Mod;
    f.a[1][1]=(1-N+Mod)%Mod,f.a[1][0]=N;
    Ans.a[0][0]=Ans.a[1][1]=1;
    for(i=0;i<len;i++)c[i]=q[i]-'0';
    for(i=0;i<len;i++){
        tmp1=Ans*Ans;
        tmp2=tmp1*tmp1;
        tmp2=tmp2*tmp2;
        Ans=tmp1*tmp2;
```

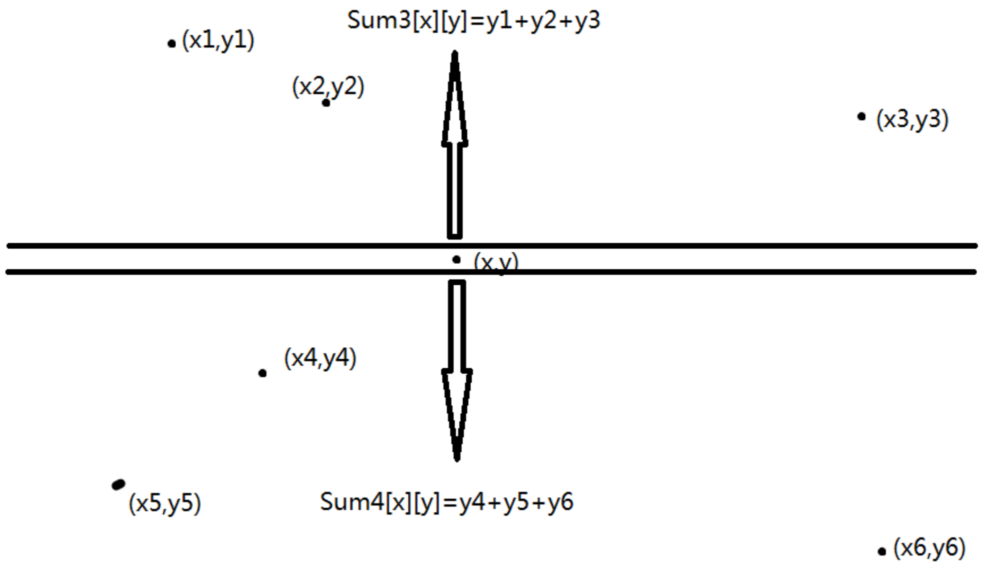
```
            Ans=Ans*(f*c[i]);
        }
        printf("%d",Ans.a[1][0]);
        return 0;
    }
```

C：异或和

题目大意：有一个$n*m$的网格图，给出小B出现在每个位置的可能性，用一个$n*m$的01矩阵表示，小B等概率出现在所有1的位置。求小A在每个位置上与小B期望曼哈顿距离的异或和，先把期望取模之后再异或。

·前缀和





将问题转化为以上两张图，求出前缀和即可$O(N^2)$求出答案。

```
#include <cstdio>
using namespace std;
long long N,M,Mod=1e9+7,Ans,Cnt,G;
long long Sum1[2005][2005][2],Sum2[2005][2005][2];
long long Sum3[2005][2005][2],Sum4[2005][2005][2];
long long Sum5[2005],Sum6[2005];
char c[2005][2005];
long long Inv(long long x,long long y) {
    long long res=1;
    while(y) {
        if(y&1)res=res*x%Mod;
        x=x*x%Mod,y>>=1;
    }
    return res;
}
int main() {
    scanf("%lld%lld",&N,&M);
    for(long long i=1; i<=N; i++)scanf("%s",c[i]+1);
    for(long long i=1; i<=N; i++)
        for(long long j=1; j<=M; j++)
            if(c[i][j]=='1')Sum5[i]++,Sum6[j]++,G++;
    G=Inv(G,Mod-2);
    for(long long i=1; i<=N; i++)
        for(long long j=1; j<=M; j++) {
            (Sum1[i][j][1]=Sum1[i][j-1][1]+Sum6[j-1])%=Mod;
            (Sum1[i][j][0]=Sum1[i][j-1][0]+Sum1[i][j][1])%=Mod;
            (Sum3[i][j][1]=Sum3[i-1][j][1]+Sum5[i-1])%=Mod;
            (Sum3[i][j][0]=Sum3[i-1][j][0]+Sum3[i][j][1])%=Mod;
        }
    for(long long i=N; i; i--)
```

```
        for(long long j=M; j; j--) {
            (Sum2[i][j][1]=Sum2[i][j+1][1]+Sum6[j+1])%=Mod;
            (Sum2[i][j][0]=Sum2[i][j+1][0]+Sum2[i][j][1])%=Mod;
            (Sum4[i][j][1]=Sum4[i+1][j][1]+Sum5[i+1])%=Mod;
            (Sum4[i][j][0]=Sum4[i+1][j][0]+Sum4[i][j][1])%=Mod;
        }
    for(long long i=1; i<=N; i++)
        for(long long j=1; j<=M; j++) {
            Cnt=0;
            Cnt=(Sum1[i][j][0]+Sum2[i][j][0]+Sum3[i][j][0]+Sum4[i][j][0])%Mod*G%Mod;
            Ans^=Cnt;
        }
    printf("%lld",Ans);
    return 0;
}
```

%%%$LowestJN$大佬