Hdu 1812 代码和注释

首先，你们需要先理解一下polya原理，直接记结论。

我感觉花个2小时看那篇论文是可以看得懂的。

这题就当成附加题吧，要成为大牛就必须做，你要相信后面还有更多更难的东西要学。

(很多东西不能畏难，懂了以后自然感觉简单，如果一开始就没信心，是很难学下去的)。

#include<cstdio>

#include<cstring>

#include<cstdlib>

#include<iostream>

#include<algorithm>

#include<cmath>

#include<set>

#include<queue>

#include<vector>

#define maxn 2001000

using namespace std;

int i,j,k,m,n;

int a[40][40],b[40][40];

typedef long long ll;

struct BigInt{ //我们没必要为了高精度特地去学java，写好了模板，每次copy一遍就好了

const static int mod = 10000;

const static int DLEN = 4;

int a[600],len;

BigInt(){

memset(a,0,sizeof(a));

len = 1;

}

BigInt(long long v){

memset(a,0,sizeof(a));

len = 0;

do{

a[len++] = v%mod;

v /= mod;

}while(v);

}

BigInt operator +(const BigInt &b)const{

BigInt res;

res.len = max(len,b.len);

for(int i = 0;i < res.len;i++){

res.a[i] += ((i < len)?a[i]:0)+((i < b.len)?b.a[i]:0);

res.a[i+1] += res.a[i]/mod;

res.a[i] %= mod;

}

if(res.a[res.len] > 0)res.len++;

return res;

}

BigInt operator -(const BigInt &b)const{

BigInt res;

int x=0;

for (int i=0;i<len;i++){

res.a[i]=a[i]-x-b.a[i];

if (res.a[i]<0)res.a[i]+=mod,x=1;else x=0;

}

res.len=len;

while (res.a[res.len-1]==0 && res.len>1)res.len--;

return res;

}

BigInt operator \*(const BigInt &b)const{

BigInt res;

for(int i = 0; i < len;i++){

int up = 0;

for(int j = 0;j < b.len;j++){

int temp = a[i]\*b.a[j] + res.a[i+j] + up;

res.a[i+j] = temp%mod;

up = temp/mod;

}

if(up != 0)

res.a[i + b.len] = up;

}

res.len = len + b.len;

while(res.a[res.len - 1] == 0 &&res.len > 1)res.len--;

return res;

}

BigInt operator / (const int b){

BigInt res;

int x=0;

for (int i=len-1;i>=0;i--){

res.a[i]=(x\*mod+a[i])/b;

x=(x\*mod+a[i])%b;

}

res.len=len;

while (res.a[res.len-1]==0&&res.len>1)res.len--;

return res;

}

void output(){

printf("%d",a[len-1]);

for(int i = len-2;i >=0 ;i--)

printf("%04d",a[i]);

printf("\n");

}

};

void inita(int n){//对n\*n的方格标上1,2,3,...n\*n的编号

int idx=0;

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

a[i][j]=++idx;

}

int getsame(int a[][40],int n){ //得到不同的等价类。

bool vis[40][40];

memset(vis,0,sizeof(vis));

int ret=0;

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

if (!vis[i][j]){

ret++;

int x=i,y=j;

while (!vis[x][y]){

vis[x][y]=1;

int axy=a[x][y];

x=axy/n+1;

y=axy%n;

if (y==0)x--,y=n;

}

}

return ret;

}

void reflection(int b[][40],int a[][40],int n){ // 反射操作，a[][]反射后，得到b[][]

int mid=n/2;

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

b[i][j]=a[i][n+1-j];

}

void rotate(int a[][40],int n){ //旋转操作，把a[][]逆时针旋转90度，放入a[][]

int b[40][40];

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

b[i][j]=a[n-j+1][i];

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

a[i][j]=b[i][j];

}

BigInt mi(int c,int k){ //快速幂求c^k，用了Bigint后，快速幂代码几乎不用改变。

BigInt ret=1;

BigInt a(c);

while (k){

if (k&1)ret=ret\*a;

a=a\*a;

k>>=1;

}

return ret;

}

BigInt calc(int n,int c){ //对n\*n的格子进行c种颜色的染色，返回染色数。

inita(n);

BigInt ret;//ll ret=0;

for (int i=0;i<=3;i++){ //执行4次，分别是旋转0度，90度，180度，270度

k=getsame(a,n);

ret=ret+mi(c,k); //得到旋转置换的染色数

reflection(b,a,n);

k=getsame(b,n);

ret=ret+mi(c,k); //得到旋转+反射置换的染色数。

rotate(a,n); //每次计算后，就逆时针旋转90度。

}

ret=ret/8; //因为有8种置换，要除以8

return ret;//ret.output();

}

int c;

BigInt ans[31][31];

int main(){

ans[1][1]=10;

for (int i=1;i<=30;i++)

for (int j=1;j<=30;j++)

ans[i][j]=calc(i,j); //先预处理出所有情况，再输出，可以节省3倍时间。

while (scanf("%d%d",&n,&c)!=EOF)

ans[n][c].output();//calc(n,c);

return 0;

}