刘牧宸

2019年8月7日

8052

首先，先画出来这道题的二维表示，y 表示1，x表示0，那么就有了(n,m)点，也就是我们最终要到达的地点，而起点自然就是(0,0)，为了满足这道题的条件，它的任何前缀字符串必须满足1的数量不能小于0的数量，所以就有C(n+m,m)种方法。

(0,0)关于y=x-1的对称点(1,-1)，从(1,-1)走到（m,n）一定与直线y=x-1相交，因为m,n在该直线的上边，1,-1在该直线的下

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
| 1，-1 |  |  |  |
|  | 0，0 |  | n,m |

所以呢合法（满足条件）的路线必然满足 y>=x的条件，那么不合法的就得满足y<x的条件，y=x-1,下面一部分取个名字叫合法路线。那么从1,-1走到m,n有多少种方法也就、是从0,0走到m,n的不合法的方法数。

合法，非法分界线

代码：

#include<iostream>

#include<algorithm>

#include<string.h>

#include<stdio.h>

using namespace std;

const int maxn=1000000;

const int mod=20100501;

bool isp[2000010];

int pp[2000010];

int len=0;

int n,m;

int t;

void ss(int n) {

for(int i=0; i<=n; i++)

isp[i]=1;

isp[0]=isp[1]=0;

for(int i=2; i<=n; i++)

if(isp[i]) {

pp[len++]=i;

for(int j=2\*i; j<=n; j+=i)

isp[j]=0;

}

}

int cal(int p,int n) {

int ans=0;

while(n) {

n/=p;

ans+=n;

}

return ans;

}

int main() {

ss(maxn\*2);

cin>>t;

while(t--) {

long long ans=1;

cin>>n>>m;

int nm=n+1-m;

for(int i=0; i<len&&pp[i]<=(n+m); i++) {

int cnt=0;

while(nm%pp[i]==0) {

nm/=pp[i];

cnt++;

}

cnt=cnt+cal(pp[i],n+m)-cal(pp[i],m)-cal(pp[i],n+1);

for(int j=1; j<=cnt; j++) {

ans=ans\*pp[i];

if(ans>=mod)

ans%=mod;

}

}

cout<<ans<<endl;

}

return 0;