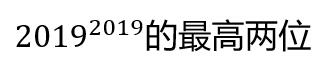
Ans:43390，直接枚举

* 

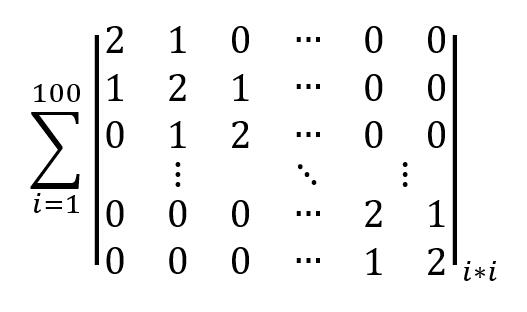
Ans:11

，，故答案为11

* 

Ans:19

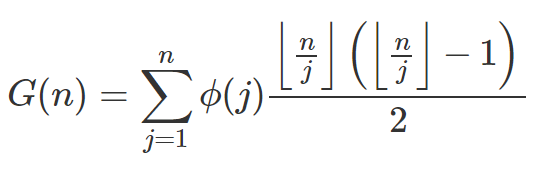
for(long double ans=1, i=2; i<=2019; i+=1){ans\*=i; while(ans>100)ans/=10;} 可得。

* 

Ans:5150

利用行列式性质可得递推式D\_n=2\*D\_(n-1)-D\_(n-2)，可得原式=n\*(n+3)/2。

Ans: 11877793747986593872

利用，再分块可得，代码如下：

#include<iostream>

#include<map>

using namespace std;

typedef unsigned long long ul;

const ul n=1ll<<40;

const int N=2e6;

ul phi[N], Phi[N], prime[N];

int tot;

map<ul, ul> Map;

ul doMul(ul a, ul b){

return (!(a&1)?(a>>1)\*b:(b>>1)\*a);

}

ul getPhi(ul n){

if(n<N)return Phi[n];

else {

if(Map[n])return Map[n];

ul ret = doMul((n+1),n), i;

ret -= (n+1)/2;

ul mid = sqrt(n)+1;

for(i=2; i<mid; i++){

ret -= Phi[i]\*(n/i-n/(i+1));

}

for(i=n/mid; i>=2; i--){

ret -= getPhi(n/i);

}

Map[n]=ret;

return ret;

}

}

int main(){

// calculate phi

phi[1]=Phi[1]=1;

for(int i=2; i<N; i++){

if(!phi[i]){

phi[i]=i-1;

prime[++tot]=i;

}

for(int j=1; j<=tot && i\*prime[j]<N; j++){

if(i % prime[j]==0){

phi[i\*prime[j]]=phi[i]\*prime[j];

break;

}

else{

phi[i\*prime[j]]=(phi[i])\*(prime[j]-1);

}

}

Phi[i]=phi[i]+Phi[i-1];

}

ul ans=0;

for(ul k=1, pre=getPhi(n); k<N; k++){

ul now=doMul(k,(k-1));

ul nxt=getPhi(n/(k+1));

ans+=now\*(pre-nxt);

pre = nxt;

}

for(ul j=1; j<=n/N; j++){

ans+=phi[j]\*doMul((n/j),(n/j-1));

}

cout<<ans;

return 0;

}

* 斐波那契数列第

，n很大的时候后项可以忽略。问题转化为求a\*b^k的前4位，利用对数的思想解决问题。