首先，很明显，杨辉三角之特点在于其****行数等于每行的数字数****。因此，可以很容易使用求和公式求出1到n行一共有多少个数字。

其次，通过观察，可以发现，奇数个数比偶数个数更有规律，其规律在于：

****1.****每行奇数个数一定为2^k（k为自然数）

****2.****当行数恰为2^k（k为自然数）时，奇数个数为2^k，偶数个数为零

****3.****当行数恰为2^k（k为自然数）时，奇数个数和恰为3^(k-1)

****4.****更巧妙的是：这个规律能更加扩展到一个不为2^k的数上，因为每一个数，都能分解为若干项2^k的和的形式。

举个例子吧：当n=2333;

****2333 = 2048+256+16+8+4+1****

通过暴力程序，我们可以找出2333的所有奇数个数为190985

****177147×1 + 6561×2 + 81×4 + 27×8 + 9×16 + 1×32 恰好等于 190985！****

对于一个大于1正整数n可以[分解质因数](https://baike.baidu.com/item/%E5%88%86%E8%A7%A3%E8%B4%A8%E5%9B%A0%E6%95%B0" \t "https://baike.baidu.com/item/%E7%BA%A6%E6%95%B0%E4%B8%AA%E6%95%B0%E5%AE%9A%E7%90%86/_blank)：

IMG_256

则n的[正约数](https://baike.baidu.com/item/%E6%AD%A3%E7%BA%A6%E6%95%B0/882466" \t "https://baike.baidu.com/item/%E7%BA%A6%E6%95%B0%E4%B8%AA%E6%95%B0%E5%AE%9A%E7%90%86/_blank)的个数就是

IMG_257

。

其中a1、a2、a3…ak是p1、p2、p3，…pk的指数。

IMG_256

1^3+2^3+3^3+...+n^3=(1+2+3+...+n)^2

对于一个大于1正整数n可以[分解质因数](https://baike.baidu.com/item/%E5%88%86%E8%A7%A3%E8%B4%A8%E5%9B%A0%E6%95%B0" \t "https://baike.baidu.com/item/%E7%BA%A6%E6%95%B0%E5%92%8C%E5%AE%9A%E7%90%86/_blank)：n=p1^a1\*p2^a2\*p3^a3\*…\*pk^ak,

则由[约数个数定理](https://baike.baidu.com/item/%E7%BA%A6%E6%95%B0%E4%B8%AA%E6%95%B0%E5%AE%9A%E7%90%86" \t "https://baike.baidu.com/item/%E7%BA%A6%E6%95%B0%E5%92%8C%E5%AE%9A%E7%90%86/_blank)可知n的正约数有(a₁+1)(a₂+1)(a₃+1)…(ak+1)个，

那么n的(a₁+1)(a₂+1)(a₃+1)…(ak+1)个正约数的和为

f(n)=(p1^0+p1^1+p1^2+…p1^a1)(p2^0+p2^1+p2^2+…p2^a2)…(pk^0+pk^1+pk^2+…pk^ak）

## **筛约数和**

t[i] 表示i的约数和

e[i] 表示i的约数中，不能被i的最小素因子整除的约数和

A、i是质数，t[i]=i+1,e[i]=1

B、i不是质数

i\*pj的最小素因子是pj

1、如果i不是pj的倍数，那么i的所有约数中，必然没有pj的倍数

可以用反证法证明这个：设x是i的约数，且x是pj的倍数，

那么 x=pj\*b，i=x\*a=pj\*b\*a

即i是pj的b\*a倍，与i不是pj的倍数相矛盾

令S表示i的约数集，S’表示i的约数翻pj倍后的数的集合

则S∩S’=∅，则S和S’中无重复元素

所以t[i\*pj]=S+S'=t[i]+t[i]\*pj=t[i]\*(pj+1)

S’中的所有元素都能整除pj，所以e[i\*pj]=t[i]

2、如果i是pj的倍数，那么S和S’必有交集T

T=S中pj的倍数

所以i\*pj的约数和要去除交集T

那么t[i\*pj]=S+S'-T=S'+S-T=t[i]\*pj+e[i]

因为pj既是i的最小素因子，有事i\*pj的最小素因子

所以e[i\*pj]=e[i]

## **筛约数个数和**

理论基础：

1、对n质因数分解，n=p1^k1 \* p2^k2 \* p3^k3 ……

则n的约数个数为(k1+1)\*(k2+1)\*(k3+1)……

2、线性筛素数时，用i和素数pj来筛掉 i\*pj，

其中pj一定是i\*pj的最小素因子

如果i是pj的倍数，pj也是i的最小素因子

设t[i] 表示i的约数个数，e[i] 表示i的最小素因子的个数

A、如果i是质数，t[i]=2,e[i]=1

B、如果i不是质数，枚举已有的质数pj

i\*pj的最小素因子是pj

1、如果i是pj的倍数那么e[i]即为i中包含的pj的个数，所以i\*pj中包含的pj的个数为e[i]+1

       所以e[i\*pj]=e[i]+1,t[i\*pj]=t[i]/(e[i]+1)\*(e[i]+2)

1. 如果i不是pj的倍数，e[i\*pj]=1,t[i\*pj]=t[i]\*t[pj]（积性函数的性质）=t[i]\*2（素数的约数个数=2）

莫比乌斯函数  
void get\_mu(int n){

int i,j,k;

mu[1] = 1;

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

{

if( !vis[i] )

{

prim.push\_back(i);

mu[i] = -1;

}

for(j=0;j<prim.size()&&i\*prim[j]<=n;j++)

{

vis[i\*prim[j]] = true;

if( i%prim[j] == 0 ) break;

else mu[i\*prim[j]] = -1\*mu[i];

}

}

}

欧拉函数

void Prim(int n){

int i,j,k;

phi[1] = 1;

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

{

if( !vis[i] ) prim.push\_back(i),phi[i]=i-1;

for(j=0;j<prim.size()&&i\*prim[j]<=n;++j)

{

vis[i\*prim[j]] = true;

if( i%prim[j] == 0 )

{

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

break;

}

else

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

}

}

}

快速幂

ll mypow(ll a,ll b,ll m){

ll inv = 1;

while( b ){

if( b&1 ) inv = inv\*a%m;

a = a\*a%m;

b>>=1;

}

return inv;

}

New：





