原创 数论模板

2019-10-18 08:26:10 _-Y-_-Y-_ 阅读数 1 文章标签: ACM 更多 编辑

gcd+ecgcd+lcm

```cpp
#include <cstdio>
#include <iostream>
using namespace std;
int gcd(int a,int b){
  return b==0?a:gcd(b,a%b);
}
int exgcd(int a,int b,int &x, int &y){
  if(b==0){ x=1,y=0;return a;}
  int t=exgcd(b,a%b,x,y);
  int x0=x,y0=y;
  x=y0,y=x0-(a/b)*y0;
  return t;
}
int lcm(int a,int b){
    return a*b/gcd(a,b);
}
```

Inv （乘法逆元）

```cpp
(a/b)%MOD=(a*inv(b,MOD))%MOD
-----------------------------------------------
#define ll long long
ll exgcd(ll a,ll b,ll &x,ll &y){
    if(a%b==0){
        x=0ll;y=1ll;
        return b;
    }
    ll v,tx,ty;
    v=exgcd(b,a%b,tx,ty);
    x=ty;
    y=tx-a/b*ty;
    return v;
}

ll inv(ll a,ll p){
    if(!a) return 0ll;
    ll x,y;
    exgcd(a,p,x,y);
    x=(x%p+p)%p;
    return x;
}
-------------------------------------------------------------
ll quick_pow(ll a,ll b,ll mod){
    ll ans=1;
    while(b)
    {
        if(b&1) ans=(ans*a)%mod;
        a=(a*a)%mod;
        b>>=1;
    }
    return ans;
}
ll inv(ll x,ll mod){
    return quick_pow(x,mod-2,mod);
}
-----------------------------------------------------------
```

卡特兰数

```
/*1
  2
  5
  14
  42
  132
  429
  1430
  4862
  16796
  58786
  208012
  742900
  2674440
  9694845
  35357670
  129644790
  477638700
  1767263190
  6564120420
  24466267020
  91482563640
  343059613650
  1289904147324
  4861946401452
  18367353072152
  69533550916004
  263747951750360
  1002242216651368
  3814986502092304
  14544636039226909
  55534064877048198
  212336130412243110
  812944042149730764
  3116285494907301262
  11959798385860453492
  45950804324621742364
  176733862787006701400
  680425371729975800390
  2622127042276492108820
  10113918591637898134020
  39044429911904443959240
  150853479205085351660700
  583300119592996693088040
  2257117854077248073253720
  8740328711533173390046320
  33868773757191046886429490
  131327898242169365477991900
  509552245179617138054608572
  1978261657756160653623774456
  7684785670514316385230816156
  29869166945772625950142417512
  116157871455782434250553845880
  451959718027953471447609509424
  1759414616608818870992479875972
  6852456927844873497549658464312
  26700952856774851904245220912664
  104088460289122304033498318812080
  405944995127576985730643443367112
  1583850964596120042686772779038896
  6182127958584855650487080847216336
  24139737743045626825711458546273312
  94295850558771979787935384946380125
  368479169875816659479009042713546950
  1440418573150919668872489894243865350
  5632681584560312734993915705849145100
  22033725021956517463358552614056949950
  86218923998960285726185640663701108500
  337485502510215975556783793455058624700
```

```
  1321422108420282270489942177190229544600
  5175569924646105559418940193995065716350
  2027689038970939986292899856825464102570 0
  7946348936507737784120823763234926888450 0
  3114968783111033211375362915188091340272 40
  1221395654430378811828760722007962130791020
  4790408930363303911328386208394864461024520
  1879314272680988457521136127908754519325004 0
  7374524361153245845969015185464732923933560 0
  2894500811752648994542838460294907672643922 30
  1136359577947336271931632877004667456667613940
  4462290049988320482463241297506133183499654740
  1752658501561677683473514051791565563639623428 0
  6885444113278019470788805203466864714298520610 0
  2705574510393951180286424632891685664206712804 40
  1063353702922273835973036658043476458723103404520
  4180080073556524734514695828170907458428751314320
  1643531483466542679706914496076288614336759039494 0
  6463326058576291437049663748614618146268153526100 0
  2542241583040007965239534407788416470865473720266 00
  1000134600800354781929399250536541864362461089950800
  3935312233584004685417853572763349509774031680023800
  1548735782249188940712832696377834323201393112783560 0
  6096087653534041575146256358082964889196972890743800 0
  2399933455180770051689157766234767230062808274882296 00
  9449737979774282078526058704549395968372307582349040 50
  3721443204405954385563870541379246659709506697378694300
  1465792935612957543701687784665703276171295495089975510 0
  5774335806960135778218770060804285633402073162475661100 0
  2275088307942293496618195403956888539560416826015410473 40
  8965199470901314966871700700741006324208375215387459093 20
*/
#include <bits/stdc++.h>
using namespace std;
int a[110][1000];
int b[1000];
void catalan(){
  a[1][0]=1;
  b[1]=1;
  int len=1;
  for(int i=2;i<=100;i++){
    for(int j=0;j<len;j++)
      a[i][j]=a[i-1][j]*(4*(i-1)+2);
    int carry=0,temp=0;
    for(int j=0;j<len;j++){
      temp=a[i][j]+carry;
      a[i][j]=temp%10;
      carry=temp/10;
    }
    while(carry){
      a[i][len++]=carry%10;
      carry/=10;
    }
    carry=0;
    for(int j=len-1;j>=0;j--){
      temp=carry*10+a[i][j];
      a[i][j]=temp/(i+1);
      carry=temp%(i+1);
    }
    while(!a[i][len-1]) len--;
    b[i]=len;
  }
}
int main(){
  catalan();
  int n;
  while(cin>>n){
    for(int i=b[n]-1;i>=0;i--){
      cout<<a[n][i];
    }
    cout<<endl;
```

```
  }
  return 0;
}

```

埃氏筛 欧拉筛

```
#include <algorithm>
#include <stdio.h>
#include <string.h>
#define maxn 10000010
using namespace std;
bool u[maxn];
int su[maxn];
int num=1;
void Es(int n){
  int i,j;
  memset(u,true,sizeof(u));
  for(i=2;i<n;i++){
    if(u[i]) su[num++]=i;
    for(j=1;j<num;j++){
      if(i*su[j]>n) break;
      u[i*su[j]]=false;
      if(i%su[j]==0) break;
    }
  }
}
void Is(int n){
  memset(u,true,sizeof(u));
  for(int i=2;i<n;i++){
    if(u[i]){
      su[num++]=i;
      for(int j=i+i;j<n;j+=i){
        u[j]=false;
      }
    }
  }
}

```

筛选法+试除法

```
#include <bits/stdc++.h>
using namespace std;
#define maxn 11000
bool u[maxn];
int su[maxn];
int a[maxn];
long long s[maxn];
int num=1;
void Es(int n){
  int i,j;
  memset(u,true,sizeof(u));
  for(i=2;i<n;i++){
    if(u[i]) su[num++]=i;
    for(j=1;j<num;j++){
      if(i*su[j]>n) break;
      u[i*su[j]]=false;
      if(i%su[j]==0) break;
    }
  }
}
bool pri(long long n){
  if(n<10000){
    return u[n];
  }else{
    for(int i=1;i<num;i++){
      if(n%su[i]==0) return false;
    }
    return true;
```

```
  }
}

```

Miller_Rabin素性测试

```
#include <bits/stdc++.h>
using namespace std;
int prime[10]={2,3,5,7,11,13,17,19,23,29};
typedef long long ll;
ll pow(ll x,ll n,ll mod){//快速幂
    ll res=1;
    while(n>0){
       if(n%2==1){
         res=res*x;
         res=res%mod;
       }
       x=x*x;
       x=x%mod;
       n>>=1;
    }
    return res;
}
ll mulit(ll a,ll b,ll c){//龟速乘
    ll ans=0;
    ll res=a;
    while(b){
      if(b&1)
        ans=(ans+res)%c;
      res=(res+res)%c;
      b>>=1;
    }
    return ans;
}
bool Miller_Rabin(int x)     //判断素数
{
    int i,j,k;
    int s=0,t=x-1;
    if(x==2)  return true;   //2是素数
    if(x<2||!(x&1))  return false;     //如果x是偶数或者是0,1，那它不是素数
    while(!(t&1))  //将x分解成(2^s)*t的样子
    {
        s++;
        t>>=1;
    }
    for(i=0;i<10&&prime[i]<x;++i)      //随便选一个素数进行测试
    {
        int a=prime[i];
        int b=pow(a,t,x);      //先算出a^t
        for(j=1;j<=s;++j)    //然后进行s次平方
        {
            k=mulit(b,b,x);   //求b的平方
            if(k==1&&b!=1&&b!=x-1)     //用二次探测判断
              return false;
            b=k;
        }
        if(b!=1)  return false;   //用费马小定律判断
    }
    return true;   //如果进行多次测试都是对的，那么x就很有可能是素数
}
int main()
{
    int x;
    scanf("%d",&x);
    if(Miller_Rabin(x))  printf("Yes");
    else  printf("No");
    return 0;
}
```

欧拉公式

```
inline ll phi(ll n) {
    if (n == 1) return 1;
    ll ans = n, m = sqrt(n);
    for (register ll i = 2; i <= m; i++)
        if (n % i == 0) {
            ans = ans / i * (i - 1);
            while (n % i == 0) n /= i;
        }
    if  (n > 1) ans = ans / n * (n - 1);
    return ans;
}

```

卢卡斯定理求组合数

```
#include<bits/stdc++.h>
#define N 100010
using namespace std;
typedef long long ll;
ll a[N];
int p;
ll pow(ll y,int z,int p){
    y%=p;ll ans=1;
    for(int i=z;i;i>>=1,y=y*y%p)if(i&1)ans=ans*y%p;
    return ans;
}
ll C(ll n,ll m){
    if(m>n)return 0;
    return ((a[n]*pow(a[m],p-2,p))%p*pow(a[n-m],p-2,p)%p);
}
ll Lucas(ll n,ll m){
    if(!m)return 1;
    return C(n%p,m%p)*Lucas(n/p,m/p)%p;
}
inline int read(){
    int f=1,x=0;char ch;
    do{ch=getchar();if(ch=='-')f=-1;}while(ch<'0'||ch>'9');
    do{x=x*10+ch-'0';ch=getchar();}while(ch>='0'&&ch<='9');
    return f*x;
}
int main(){//C(n+m,n)
    int T=read();
    while(T--){
        int n=read(),m=read();p=read();
        a[0]=1;
        for(int i=1;i<=p;i++)a[i]=(a[i-1]*i)%p;
        cout<<Lucas(n+m,n)<<endl;
    }
}

```

费马小定理+快速幂求组合数

```
#include <bits/stdc++.h>
#define mod 1000000007
typedef long long LL;
const int maxn = 100000 + 10;
LL fac[maxn];//乘阶表
LL power(LL a,LL b){//快速幂
    a%=mod;
    LL ans = 1;
    while(b){
        if(b&1)ans = (ans*a)%mod;
        b>>=1;
        a = (a*a)%mod;
    }
    return ans;
}
LL inv(LL a){//返回逆元（费马小定理）
```

```
    return power(a,mod-2)%mod;
}
void solve(){//计算乘阶表
    fac[0] = 1;
    for(int i = 1;i<=maxn-1;i++){
        fac[i] = (fac[i-1]*i)%mod;
    }
}
LL comb(int n,int k){//返回组合数（组合数公式）
    if(k>n)return 0;
    return (fac[n]*inv(fac[k])%mod*inv(fac[n-k])%mod)%mod;
}
int main()
{
    solve();
    int n,k;
    while(scanf("%d%d",&n,&k)!=EOF){
        printf("%lld\n",comb(n,k));
    }
    return 0;
}
```

欧拉函数求与n互质的个数

```
#include <cstdio>
int oula(int n){
    int ou=n;
    for(int i=2;i*i<=n;i++){//n/2是最大的因子
        if(n%i==0){
            ou-=ou/i;//欧拉函数 φ(n)
            while(n%i==0){//除去非质数的因子
                    n=n/i;
            }
        }
    }
    if(n>1){//剩余的n可能也是质因子
        return ou-=ou/n;
    }
    return ou;
}
int main() {
    int n;
    while(~scanf("%d", &n)&&n){
        printf("%d\n", oula(n));
    }
    return 0;
}
```

文章最后发布于: 2019-10-18 08:31:51