数学基础

1. 等差数列

1. $a_n = a_1 + (n-1) \times d$
2. $S_n = na_1 + \frac{n(n-1)d}{2}, n \in N^*$

其中，a_1 为首项，a_n 为末项，公差为d，前n项和为S_n。

2. 等比数列

1. $a_n = a_1 \cdot q^{(n-1)}$
2. $S_n = \frac{a_1(1-q^n)}{1-q}(q\neq1)$

其中，a_1 为首项，a_n 为末项，公比为d，前n项和为S_n。

3. 取模运算

3.1 四则运算（除法除外）

1. $(a + b) \% p = (a \% p + b \% p) \% p$
2. $(a – b) \% p = (a \% p – b \% p) \% p$
3. $(a * b) \% p = (a \% p * b \% p) \% p$
4. $(a^b) \% p = ((a \% p)^b) \% p$

3.2 结合律

1. $((a + b) \% p + c) \% p = (a + (b+c) \% p) \% p$
2. $((a \times b) \% p \times c) \% p = (a \times (b \times c) \% p) \% p$

3.3 交换律

1. $(a + b) \% p = (b+a) \% p$
2. $(a \times b) \% p = (b \times a) \% p$

3.4 分配律

1. $((a + b) \% p \times c) \% p = ((a \times c) \% p + (b \times c) \% p) \% p$

3.5 重要定理

1. 若$a≡b (\% p)$，则对于任意的c，都有$(a + c) ≡ (b + c) (\%p)$
2. 若$a≡b (\% p)$，则对于任意的c，都有$(a \times c) ≡ (b \times c) (\%p)$
3. 若$a≡b (\% p) ，c≡d (\% p)$，则
   1. $(a + c) ≡ (b + d) (\%p)$
   2. $(a – c) ≡ (b – d) (\%p)$
   3. $(a \times c) ≡ (b \times d) (\%p)$
   4. $(a / c) ≡ (b / d) (\%p)$

4. 幂指

$ans=base^{power}$，例如$2^{1000000}$

4.1 递推

```
long long power1(long long base,long long power,long long mod){
```

```
    long long result = 1;

    for(int i=1;i<=power;i++)
        result = ((result%mod)*(base%mod))%mod;

    return result;
}
```

4.2 快速幂

1. $3^{10}=3\times3\times3\times3\times3\times3\times3\times3\times3\times3$
2. $3^{10}=(3\times3)\times(3\times3)\times(3\times3)\times(3\times3)\times(3\times3)$
3. $3^{10}=(3\times3)^5$
4. $3^{10}=9^{5}$
5. $9^{5}=9^{4}\times9^{1}$
6. $9^{5}=81^{2}\times9^{1}$
7. $9^{5}=5661^{1}\times9^{1}$

```
long long power2(long long base,long long power,long long mod){
    long long result = 1;

    while(power>0){
        if(power%2==0){
            power = power/2;
            base = ((base%mod)*(base%mod))%mod;
        }else{
            power = power -1;
            result = ((result%mod)*(base%mod))%mod;
            power = power/2;
            base = ((base%mod)*(base%mod))%mod;
        }
    }
    return result;
}
```

```
long long power3(long long base,long long power,long long mod){
    long long result = 1;

    while(power>0){
        if(power%2==1){
            result = ((result%mod)*(base%mod))%mod;
        }
        power = power/2;
        base = ((base%mod)*(base%mod))%mod;
    }

    return result;
}
```

```
long long power4(long long base,long long power,long long mod){
    long long result = 1;

    while(power>0){
        if(power&1){
            result = ((result%mod)*(base%mod))%mod;
        }
        power >>=1;
        base = ((base%mod)*(base%mod))%mod;
    }

    return result;
}
```

5. 约数个数和约数和

5.1 基本定理

任何大于1的整数A都可以分解成若干质数的乘积。如果不考虑这些质数次序，A可以写成标准分解式：
$A=P_1^{a1} \times P_2^{a2} \times P_3^{a3} \times \cdots \times P_n^{an}$，
其中 $P_1 \lt P_1 \lt P_1 \lt \ldots \lt P_n$ 为质数，a_i为非负整数$i = 1,2,\cdots,n$。

5.2 约数个数公式

若 $A=P_1^{a1} \times P_2^{a2} \times P_3^{a3} \times \cdots \times P_n^{an}$ 为标准公式，则A的所有约数（包括1和本身）的个数等于：
$(a_1+1)\times (a_2+1)\times (a_3+1) \times \cdots \times (a_n+n)$

5.3 约数和公式

若 $A=P_1^{a1} \times P_2^{a2} \times P_3^{a3} \times \cdots \times P_n^{an}$ 为标准公式，则A的所有约数（包括1和本身）的和等于：
$(1+p_1+p_1^2+\cdots+p_1^{a1}) \times (1+p_2+p_2^2+\cdots+p_2^{a2}) \times (1+p_3+p_3^2+\cdots+p_3^{a3}) \times \cdots \times (1+p_n+p_n^2+\cdots+p_n^{an})$

1. 递推求和O(n)

```
#include <iostream>
using namespace std;

int a1=1,q,n,sum=1,mod=9901,acc=1;

int main(){

    cin >> q >> n;
    for(int i=1;i<=n;i++){
        acc=((acc%mod)*q)%mod;
        sum = (sum%mod + acc%mod)%mod;
    }
    cout << sum << endl;

    return 0;
}
```

2. 分治求和$O(\log^2 n)$

    1. n为奇数时
       $(1+p_1+p_1^2+p_1^3+p_1^4+p_1^5+p_1^6+p_1^7)\%9901$
       $=(1+p_1+p_1^2+p_1^3+p_1^4(1+p_1+p_1^2+p_1^3))\%9901$
       $=\underbrace{(1+p_1+p_1^2+p_1^3)} \times \underbrace{(1+p_1^4)}\%9901$
    2. n为偶数时
       $(1+p_1+p_1^2+p_1^3+p_1^4+p_1^5+p_1^6)\%9901$
       $=(1+p_1+p_1^2)+p_1^3+p_1^4(1+p_1+p_1^2))\%9901$
       $=(\underbrace{(1+p_1+p_1^2)} \times \underbrace{(1+p_1^4)}+\underbrace{p_1^3})\%9901$

```
#include <iostream>
#include "power4.h"
using namespace std;

long long sum(long long a, long long n,long long mod){
    if(n ==1) return a;
    long long t = sum(a,n/2,mod);
    if(n&1){
        long long cur = power4(a,n/2+1,mod);
        t = (t+t*cur%mod)%mod;
        t = (t+cur)%mod;
    }else{
        long long cur = power4(a,n/2,mod);
        t=(t+t*cur%mod) % mod;
```

```
    }
    return t;
}

int main(){

    cout << 1+sum(2,3,9901);

    return 0;
}
```

5.4 分解质因数

```
#include <iostream>
using namespace std;

int main(){
    int n,i=2;

    cin >> n;
    cout << n << "=";
    do{
        while(n%i==0){
            cout << i;
            n/=i;
            if(n!=1)
                cout << "*";
        }
        i++;
    }while(n!=1);
    cout <<endl;

    return 0;
}
```

5.5 POJ1845 Sumdiv

Consider two natural numbers A and B. Let S be the sum of all natural divisors of \(A^B \).

Input

The only line contains the two natural numbers A and B, (0 <= A,B <= 500000000) separated by blanks.

Output

The only line of the output will contains S modulo 9001.

Sample Input

```
2 3
```

Sample Output

```
15
```

HINT

\(2^3 = 8 \)

The natural divisors of 8 are: 1,2,4,8. Their sum is 15.

15 modulo 9901 is 15 (that should be output).

Analysis

\((1+p_1+p_1^2+\cdots+p_1^{a_1B}) \times (1+p_2+p_2^2+\cdots+p_2^{a_2B}) \times (1+p_3+p_3^2+\cdots+p_3^{a_3B}) \times \cdots \times (1+p_n+p_n^2+\cdots+p_n^{a_nB}) \)

Code

```
#include<iostream>
#include<cstdio>
#define size 10000
#define mod 9901
using namespace std;

int p[size],n[size];

//快速幂
long long pow3(long long int a,long long int b ){
    long long int r = 1, base = a;
    while( b != 0 ){
        if( b & 1 )
            r =(r * base) % mod;
        base =(base * base) % mod;
        b >>= 1;
    }
    return r;
}

//二分递归求解等比数列之和
long long sum(long long p,long long n){
    if(n==0)
        return 1;
    if(n%2)
        return ((sum(p,n/2) % mod) * (1+pow3(p,n/2+1))% mod )% mod;
    else
        return (((sum(p,n/2-1) % mod) * (1+pow3(p,n/2+1))% mod ) % mod + pow3(p,n/2) %mod)%mod;
}

int main(){
    int A,B;
    scanf("%d%d",&A,&B);
        int js=0;
        //质因数分解A，根号法和递归法
        for(int i=2;i*i<=A;){
            if(A%i==0){
                p[js]=i;   //质因数
                n[js]=0;
                while(A%i==0){  //质因数次数
                    n[js]++;
                    A/=i;
                }
                js++;
            }
            if(i==2) i++;
            else i+=2;
        }
        //A本身就是素数的
        if(A!=1){
            p[js]=A;
            n[js++]=1;
        }

        long long ans=1;
        for(int i=0;i<js;i++){
            ans=( ans * sum(p[i],n[i]*B)%mod )%mod;
        }
```

```
        cout<<ans<<endl;

    return 0;
}
```