1 约数

约数：若整数n除以整数d的余数为0，即d能整除n，则称d是n的约数，n是d的倍数，记作d|n。

整数惟一分解定理：

- 若整数$N \geq 2$，那么 $N = p_1^{r_1} p_2^{r_2} \dots p_k^{r_k}$ (p_i为素数，$r_i \geq 0$)
- N 的正约数集合为：$\{p_1^{b_1} p_2^{b_2} \dots p_k^{b_k}\}$ $(0 \leq b_i \leq r_i)$
- N 的正约数个数为：

$$(r_1 + 1) * (r_2 + 1) * \cdots * (r_k + 1) = \prod_{i=1}^{k} (r_i + 1)$$

- 除了完全平方数，约数总是成对出现的，即 $d \leq \sqrt{N}$ 和 $\frac{N}{d} \leq \sqrt{N}$ 都是 N 的约数。

整数惟一分解定理的推论

- 除了完全平方数，约数总是成对出现的，即 $d \le \sqrt{N}$ 和 $\frac{N}{d} \le \sqrt{N}$ 都是 N 的约数。
- N 的约数个数上界为 $2\sqrt{N}$ ，时间复杂度为 $O(\sqrt{N})$ 。
- N 的所有正约数的和为：

$$(1 + p_1 + p_1^2 + \cdots + p_1^{r_1}) * \cdots * (1 + p_k + p_k^2 + \cdots + p_k^{r_k}) = \prod_{i=1}^{k} (\sum_{j=0}^{r_i} (p_i)^j)$$

$$\geq$$

[1315] 试除法求约数

给定 n 个正整数 ai，对于每个整数 ai，请你按照从小到大的顺序输出它的所有约数。

输入第一行包含整数 n。 接下来 n 行，每行包含一个整数 ai。

输出共 n 行，其中第 i 行输出第 i 个整数 ai 的所有约数。 $1 \leqslant n \leqslant 100$ $2 \leqslant ai \leqslant 2 \times 10^9$

输入样例：

```
2
6
8
```

输出样例：

```
1 2 3 6
1 2 4 8
```

分析

一个数的约数一定是成对出现的，比如d可以整除n，那么(n/d)也可以整除n。

是枚举约数，如果是那么push进栈里面，再考虑一下边界条件即可。

```
if(n%i==0)//如果i是n的一个约数
{
    res.push_back(i);
    if(i!=n/i)res.push_back(n/i);
  //考虑边界情况,有可能n是i的平方
}
```

。

```cpp
vector<int> get_divisors(int n)
{
    vector<int> res;

    for(int i=1;i<=n/i;i++)//从小到大枚举较小的那个约数即可
    {
        if(n%i==0)//如果i是n的一个约数
        {
            res.push_back(i);
            if(i!=n/i)res.push_back(n/i);
            //考虑边界情况,有可能n是i的平方
        }
    }
    sort(res.begin(),res.end());
    return res;
}
```

```
1   #include<iostream>
2   #include<algorithm>
3   #include<vector>
4   using namespace std;
5
6   vector<int> get_divisors(int n)
7   {
22
23  int main()
24  {
25      int n;
26      cin>>n;
27      while(n--)
28      {
29          int x;
30          cin>>x;
31          auto res=get_divisors(x);
32          for(auto t:res) cout<<t<<' ';
33          cout<<endl;
34      }
35      return 0;
36  }
```

[1316] 约数个数

给定 n 个正整数 ai，请你输出这些数的乘积的约数个数，答案对 10^9+7取模。

输入第一行包含整数 n。 接下来 n 行，每行包含一个整数 ai。

输出一个整数，表示所给正整数的乘积的约数个数，答案需对 109+7 取模。 $1 \leqslant n \leqslant 100$ $2 \leqslant ai \leqslant 2 \times 10^9$

输入样例：	输出样例：
3 2 6 8	12

分析：推导约数个数

$$N = p_1^{\alpha_1} * p_2^{\alpha_2} * \cdots * p_k^{\alpha_k}$$

$$ans = (\alpha_1 + 1)(\alpha_2 + 1)\ldots(\alpha_k + 1)$$

因为任何一个约数d可以表示成

$$d = p_1^{\beta_1} * p_2^{\beta_2} * \cdots * p_k^{\beta_k}, 0 \leq \beta_i \leq \alpha_i$$

每一项的β_i如果不同，那么约数d就不相同

所以n的约数就跟β_i的选法是一一对应的

β_1一共有$0 \sim \alpha_1$种选法

β_2一共有$0 \sim \alpha_2$种选法

$\ldots$

β_k一共有$0 \sim \alpha_k$种选法

乘法原理，一共有ans个约数

分析：推导约数个数

$36=2^2\times3^2$

36的因子为：

$2^0\times3^0$,$2^0\times3^1$，$2^0\times3^2$

$2^1\times3^0$,$2^1\times3^1$,$2^1\times3^2$

$2^2\times3^0$,$2^2\times3^1$,$2^2\times3^2$

共(2+1)×(2+1)=9种

$$N = p_1^{\alpha_1} * p_2^{\alpha_2} * \cdots * p_k^{\alpha_k}$$
$$ans = (\alpha_1 + 1)(\alpha_2 + 1)\ldots(\alpha_k + 1)$$

int 范围内约数个数最多的约1500多个

```cpp
#include <bits/stdc++.h>
using namespace std;
typedef long long LL;
const int mod = 1e9 + 7;
int hash1[500010];
```

```cpp
int main(){
    int n,x;
    LL ans = 1;
    cin >> n;
    while(n--){
        cin >> x;
        for(int i = 2;i <= x/i; ++i){
            while(x % i == 0){
                x /= i;
                hash1[i] ++;
            }
        }
        if(x > 1) hash1[x] ++;
    }
    for(int i=2;i<=500000;i++)
        if(hash1[i]!=0) ans = ans*(hash1[i]+1) % mod;
    cout << ans;
    return 0;
}
```

```
 6  map<int, int> tb;
 7  void getPrimes(int a) {
 8      for (int i = 2; i <= a / i; i++) {
 9          if (a % i == 0) {
10              int s = 0;
11              while (a % i == 0) {
12                  a /= i;
13                  s++;
14              }
15              tb[i] += s;
16          }
17      }
18      if (a != 1) tb[a]++;
19  }
```

```
21  int main() {
22      int n;
23      cin >> n;
24      while (n--) {
25          int a;
26          cin >> a;
27          getPrimes(a);
28      }
29      LL res = 1;
30      for (map<int, int>::iterator it = tb.begin(); it != tb.end(); it++) {
31          int k = it->first;
32          int v = it->second;
33          res *= (v + 1ll);
34          res %= MOD;
35      }
36      cout << res << endl;
37      return 0;
38  }
```

[1317] 约数之和

给定 n 个正整数 ai，请你输出这些数的乘积的约数之和，答案对 109+7取模。
输入第一行包含整数 n。 接下来 n 行，每行包含一个整数 ai。
输出一个整数，表示所给正整数的乘积的约数个数，答案需对 109+7 取模。 $1 \leqslant n \leqslant 100$ $2 \leqslant ai \leqslant 2 \times 10^9$

输入样例：
3
2
6
8

输出样例：
12

则n的约数的和(sum)：

$$sum(n) = (a_1^0 + a_1^1 + \ldots + a_1^{\alpha_1}) * (a_2^0 + a_2^1 + \ldots + a_2^{\alpha_2}) * \ldots * (a_k^0 + a_k^1 + \ldots + a_k^{\alpha_k})$$

证明：

若 n 可以分解质因数：$n = p_1^{a_1} * p_2^{a_2} * \cdots * p_k^{a_k}$，

可知，$p_1^{a_1}$ 的约数有：$p_1^0, p_1^1, p_1^2, \ldots, p_1^{a_1}$

$\ldots$

同理可知，$p_k^{a_k}$ 的约数有：$p_k^0, p_k^1, p_k^2, \ldots, p_k^{a_k}$；

实际上 n 的约数是在 $p_0^{a_0}, p_1^{a_1}, p_2^{a_2}, \ldots, p_k^{a_k}$ 每一个的约数中分别挑一个相乘得来，

可知共有 $(a_1 + 1)(a_2 + 1) \cdots (a_k + 1)$ 种挑法，即约数个数。

有 乘法原理 可知它们的和为

$$f(n) = (p_1^0 + p_1^1 + p_1^2 + \cdots + p_1^{a_1})(p_2^0 + p_2^1 + p_2^2 + \cdots + p_2^{a_2}) \cdots (p_k^0 + p_k^1 + p_k^2 + \cdots + p_k^{a_k})$$

例题

$$180 = 2^2 * 3^2 * 5^1$$

约数个数：$(2+1)(2+1)(1+1) = 18$

约数和：$(1+2+4)(1+3+9)(1+5) = 546$

$60=2^2*3^1*5^1$

约数的和：（1+2+4）*(1+3)*(1+5)=7*4*6=168

（$2^0+2^1+2^2$）*(3^0+3^1)*(5^0+5^1)

1 2 3 4 5 6 10 12 15 20 30 60 = 168

$60=2^2*3^1*5^1$

约数的和：（1+2+4）*(1+3)*(1+5)=7*4*6=168

$(2^0+2^1+2^2)*(3^0+3^1)*(5^0+5^1)$

从3个括号中任意选一个组成一个项式求和：

1 2 3 4 5 6 10 12 15 20 30 60 = 168

1： $2^0\ 3^0\ 5^0$ (000)

2： $2^1\ 3^0\ 5^0$ (100)

3： $2^0\ 3^1\ 5^0$ (010)

4： $2^2\ 3^0\ 5^0$ (200)

5： $2^0\ 3^0\ 5^1$ (001)

6： $2^1\ 3^1\ 5^0$ (110)

10 12 15 20 30 60 = 168

60： $2^2\ 3^0\ 5^1$ (211)

30： $2^1\ 3^1\ 5^1$ (111)

20： $2^2\ 3^0\ 5^1$ (201)

15： $2^2\ 3^0\ 5^0$ (011)

12： $2^2\ 3^1\ 5^0$ (210)

10： $2^1\ 3^0\ 5^1$ (101)

$$f(n) = (p_1^0 + p_1^1 + p_1^2 + \cdots + p_1^{a_1})$$

t=1

$t=t*p+1=p^1+1$

$t=t*p+1=(p^1+1)*p+1=p^2+p^1+1$

.....

$t=t*p+1=p^{a1}+....+p^2+p^1+1$

```
while(a1--){
    t=(t*p+1);
}
```

```cpp
#include<bits/stdc++.h>
using namespace std;
typedef long long LL;
const int mod=1e9+7;
int primes[5000005];
```

```cpp
//存储所有的底数和指数
int main()
{
    int n;
    cin>>n;
    while(n--)
    {
        int x;
        cin>>x;

        for(int i=2;i<=x/i;i++)
            while(x%i==0)
            {
                x/=i;
                primes[i]++;//i的质因数指数+1
            }
            if(x>1)primes[x]++;
    }
```

```
27      LL ans=1;
28      for(int i=2;i<=500000;i++){
29          int p=primes[i];
30          LL t=1;
31          if(p!=0) {
32              while(p--){
33                  t = (t*i+1) % mod;
34              }
35          }
36          ans=ans*t%mod;
37      }
38      cout<<ans<<endl;
39      return 0;
40  }
```

```cpp
const int mod = 1e9 + 7;
unordered_map<int, int> primes;
int main()
{
    int n;
    cin >> n;
    while (n --) //将每个数的质因数个数用哈希表存储
    {
        int x;
        cin >> x;
        for (int i = 2; i <= x / i; i ++)
        {
            while (x % i == 0) //每次将质因数i除干净
            {
                primes[i] ++;
                x /= i; //每次需要除掉已经统计的质因子
            }
        }
        if (x > 1) primes[x] ++; //x含大于√ x的质因数
    }
```

```
25      long long res = 1;
26      for (auto u : primes)
27      {
28          long long val = u.first, num = u.second;
29          long long p = 1;
30          while (num --) p = (p * val + 1) % mod;
31          //该项可能取值之和:val^0 + val^1 +...+ val^num
32
33          res = res * p % mod; //将每一项的可能取值连乘
34      }
35      cout << res << endl;
36  }
```

2 最大公约数最小公倍数

一般地，设$a_1, a_2, a_3, \ldots a_k$ 是k个非零整数，如果存在要给非零整数d, 使得d$|a_1$， d$|a_2, \ldots$ d$|a_k$,

那么d 就称为 $a_1, a_2, \ldots a_k$ 的公约数。

公约数中最大的一个就被称为最大公约数，记作

$\mathrm{GCD}(a_1, a_2 \ldots a_k)$

如果存在要给非零整数d, 使得$a_1|$d, $a_2|$d, $\ldots a_k|$d， 那么d就称为 a_1， $a_2, \ldots a_k$。 的公倍数。

公倍数中最小的一个就被称为最小公倍数，记作

$\mathrm{LCM}(a_1, a_2 \ldots a_k)$

2.1 最辗转相除法

辗转相除法用求两个数的最大公约数，又称为欧几里得算法，其原理就是：

$$\mathrm{GCD}(x, y) = \mathrm{GCD}(x, y - x)$$

```
int GCD(int x, int y){
    return y==0?x:GCD(y,x%y);
}
```

2.2 二进制优化

因为高精度除法不容易实现，需要做高精度运算时可以考虑二进制算法，提高GCD的效率：可以通过不断去除因子2来降低常数，这就是“二进制优化”。

若x=y,则GCD(x,y)=x,否则：

1）若x,y均为偶数，则 GCD(x,y)=GCD(x/2,y/2)

2）若x,为偶数，y为奇数则 GCD(x,y)=GCD(x/2,y)

3）若x为奇数，y为偶数GCD(x,y)=GCD(2,y/2)

4）若x,y均为奇数，则GCD(x,y)=GCD(x-y,y)

```
 5  int FASTGCD(int x,int y)
 6  {
 7      int i,j;
 8      if(x==0) return y;
 9      if(y==0) return x;//去掉所有的2
10      for(i=0;0==(x&1);++i) x>>=1;
11      for(j=0;0==(y&1);++j) y>>=1;
12      if(j<i) i=j;
13      while(1)
14      {              //若x<y，交换x,y
15          if(x<y) x^=y,y^=x,x^=y;
16          if(0==(x-=y)) return y<<i;
17          while(0==(x&1)) x>>=1;
18      }
19  }
```

[3246] Hankson的趣味题

Hanks博士是BT（Bio-Tech，生物技术）领域的知名专家，他的儿子名叫Hankson。现在，刚刚放学回家的Hankson正在思考一个有趣的问题。

今天在课堂上，老师讲解了如何求两个正整数c1和c2的最大公约数和最小公倍数。现在Hankson认为自己已经熟练地掌握了这些知识，他开始思考一个“求公约数”和“求公倍数”之类问题的“逆问题”，这个问题是这样的：已知正整数a0，a1，b0，b1，设某未知正整数x满足：

1. x和a0的最大公约数是a1；
2. x和b0的最小公倍数是b1。

Hankson的“逆问题”就是求出满足条件的正整数x。但稍加思索之后，他发现这样的x并不唯一，甚至可能不存在。因此他转而开始考虑如何求解满足条件的x的个数。请你帮助他编程求解这个问题。

[3246] Hankson的趣味题

每组输入数据的第一行为一个正整数n，表示有n组输入数据。接下来的n行每行一组输入数据，为四个正整数a0，a1，b0，b1，每两个整数之间用一个空格隔开。输入数据保证a0能被a1整除，b1能被b0整除。

每组输出共n行。每组输入数据的输出结果占一行，为一个整数。

对于每组数据：若不存在这样的x，请输出0；

若存在这样的x，请输出满足条件的x的个数。

下面是对样例数据的解释：

第一组输入数据，x可以是9、18、36、72、144、288，共有6个。

第二组输入数据，x可以是48、1776，共有2个。

[3246] Hankson的趣味题

输入样例

2

41 1 96 288

95 1 37 1776

输出样例

6

2

对于100%的数据，保证有1≤a0，b1，b0，b1≤2,000,000,000且n≤2000。

分析1

$gcd(x, a0) = a1;$
$lcm(x, b0) = b1,$
可知a1 <= x <= b1,
可以枚举X, 求出此时的
$a1' = gcd(X, a0)$和
$b1' = lcm(X, b0),$
然后判断a1'与a1是否相等
且b1'与b1是否相等。

50分

```
3  int gcd(int x, int y){
4      if(x%y==0)
5          return y;
6      return gcd(y,x%y);
7  }
8  int main(){
9      int n,a0,a1,b0,b1,d,d1;
10     cin>>n;
11     for(int i=1;i<=n;i++){
12         int ans=0;
13         cin>>a0>>a1>>b0>>b1;
14         for(int x=a1;x<=b1;x++){
15             d=gcd(a0,x);
16             d1=x/gcd(b0,x)*b0;
17             if((d==a1)&&(d1==b1))
18                 ans++;
19         }
20         cout<<ans<<endl;
21     }
22     return 0;
23 }
```

分析2

$$gcd(x, a0) = a1$$
$$lcm(x, b0) = b1$$

令$b0 = T * m$,

$X = T * q$,

$gcd(m, n) = 1$,

推出 $b1 = T * m * n$

$$\frac{b1}{b0} = \frac{T*m*q}{T*m} = q$$

```
12      int n,a0,a1,b0,b1;
13      cin>>n;
14      while(n--){
15          int t,tt,ans=0,p,q;
16          cin>>a0>>a1>>b0>>b1;
17          q=b1/b0;
18          for(int x=q;x<=b1;x=x+q){
19              t=gcd(x,a0);
20              tt=gcd(x,b0);
21              p=x/tt*b0;
22              if((t==a1)&&(p==b1))ans++;
23          }
24
25          cout<<ans<<endl;
```

q为X的一个因子，因此在遍历x时可以直接遍历q的倍数即可

p就是最小公倍数，最小公倍数等于最大公约数乘以q

70分

分析3

$$\gcd(x, a0) = a1 \quad (1)$$
$$\mathrm{lcm}(x, b0) = b1 \quad (2)$$

由(2)式： $b0 * x = b1 * gcd(x, b0)$

$$x = \frac{b1}{b0} * gcd(x, b0)$$

$$i = gcd(x, b0) \in [1, \sqrt{b0}]$$

判断：

$x = \frac{b1}{b0} * i$ 和 $x = \frac{b1}{b0} * \frac{b0}{i}$ 是否满足条件