MC0406 最小公倍数

题目链接:

https://www.matiji.net/exam/brushquestion/6/4693/305EE97B0D5E361DE6A28CD18C929AF0

题目描述:

唐僧师徒一路西行，忽然狂风大作，烈焰冲天，四周温度骤然上升！沙僧低声道:“师父，妖气冲天，怕是红孩儿来了!”果然，前方火光四射，一个身穿红衣的孩童踏火而来，正是红孩儿！他大笑道:“想要活着过此地？必须先接下我的三昧真火试炼!”

话音未落，天空中浮现出 n 团炽热的火焰，每团火焰都有一个独立燃烧周期 a1，a2，…，an，表示它们在经过该时间后会重新燃起，形成不灭的循环！ 红孩儿冷笑道:“这些火焰不会自行熄灭，只有找到一个它们共同的熄灭时刻，才能让它们同时熄灭!”

沙僧皱眉道:“这妖火威力无穷，若不及时熄灭，我们怕是难以抵挡。”

红孩儿继续冷笑道:“即便你算得出结果，还需要对 998244353 取模，否则火焰依旧不灭!”

沙僧擦了擦额头的汗:“小码哥，咱们的生死可全指望你了，快算算共同熄灭的时刻吧!”

输入格式:

第一行一个整数 n(1≤n≤104)。

第二行 n 个整数 a1~an(1≤ai≤109)。

输出格式:

一行一个整数，表示答案。

样例:

输入: 2

1 2

输出: 2

题目解析:

0、最大公因数(gcd),最小公倍数(lcm)。

1、找出所有火焰同时熄灭的时间即为找出所有熄灭时间的最小公倍数 lcm(a1……an)，要找出所有数的 lcm 只需从头到尾遍历两两求最小公倍数:

lcm(a1,a2,a3,……,an)=lcm(a1,lcm(a2,lcm(a3,……,an)))

2、对于任意两个数 x,y:

1、x 和 y 都可以分成若干个质数的积

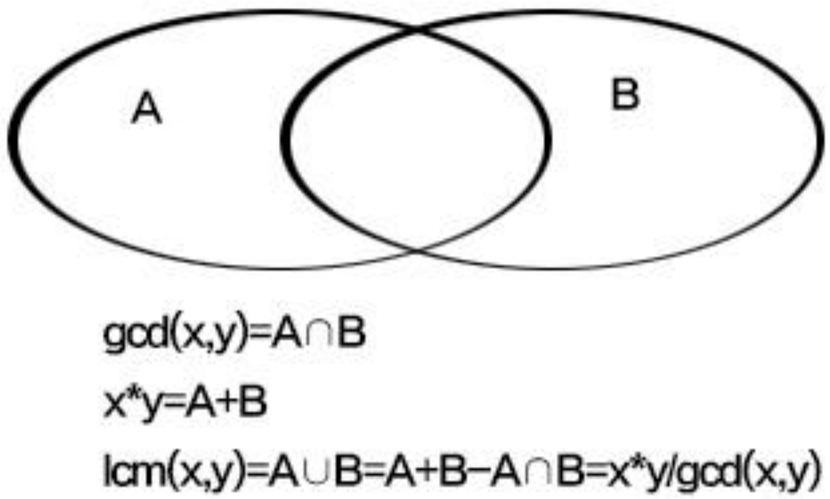
eg. 6=2*3,8=2*2*2……

2、Gcd(x,y)为 x 和 y 的乘数中公共部分的积

Eg. Gcd(6,8)=2,gcd(21,14)=7……

3、lcm(x,y)为 x 和 y 的乘数中 gcd(x,y)、x 和 y 中不同部分的积

4、x*y=lcm(x,y)*gcd(x,y)



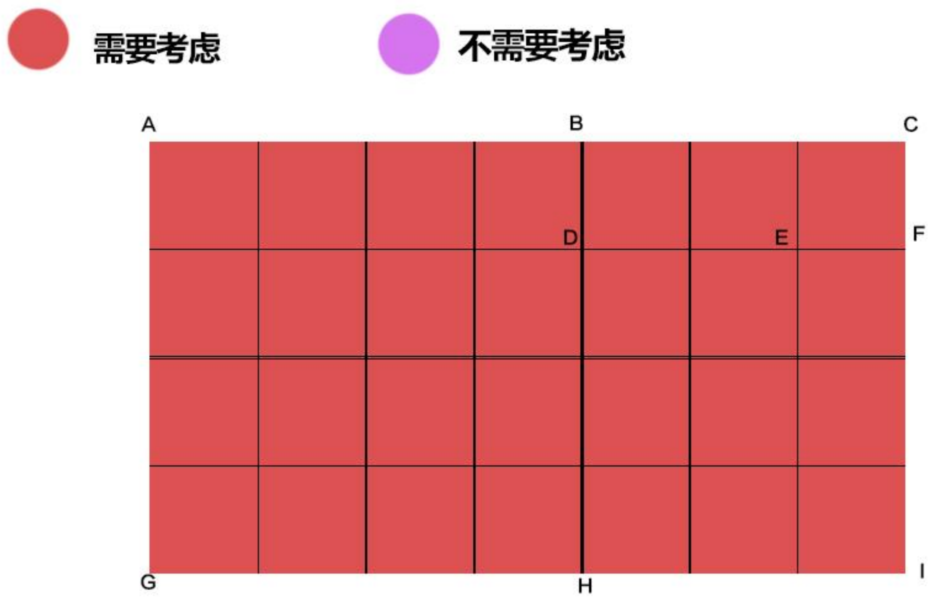
由此可推理出解题方法:

辗转相除法：先求出两个数 a,b 的 gcd(a,b)，再用两个数之积 a*b

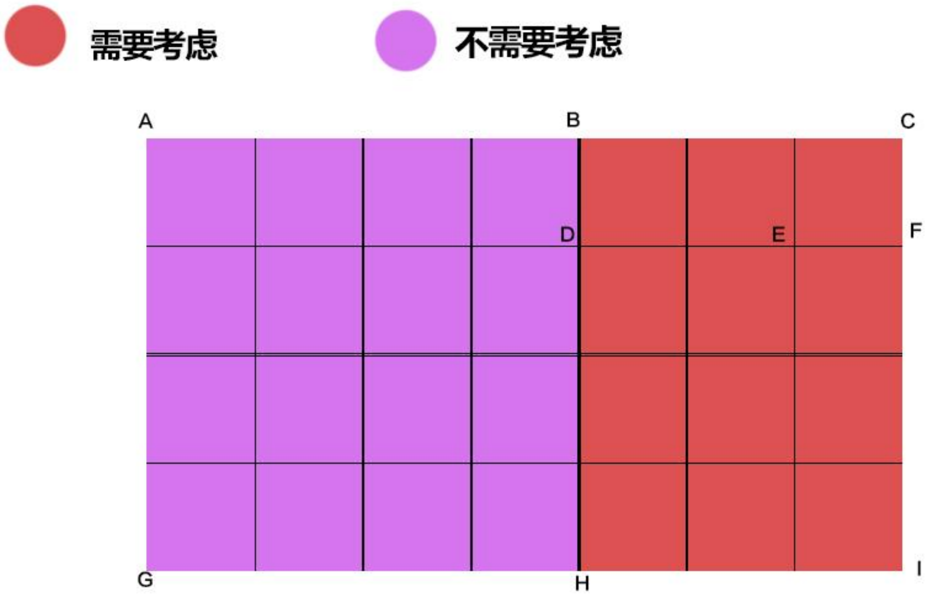
除以 gcd(a,b)，再算 lcm(lcm(a,b),c)以此类推

将 a*b 看作是长为 a，宽为 b 的长方形的面积，gcd(a,b)即为若干个大小相同的的正方形恰好填满长方形时，正方形面积最大时的边长

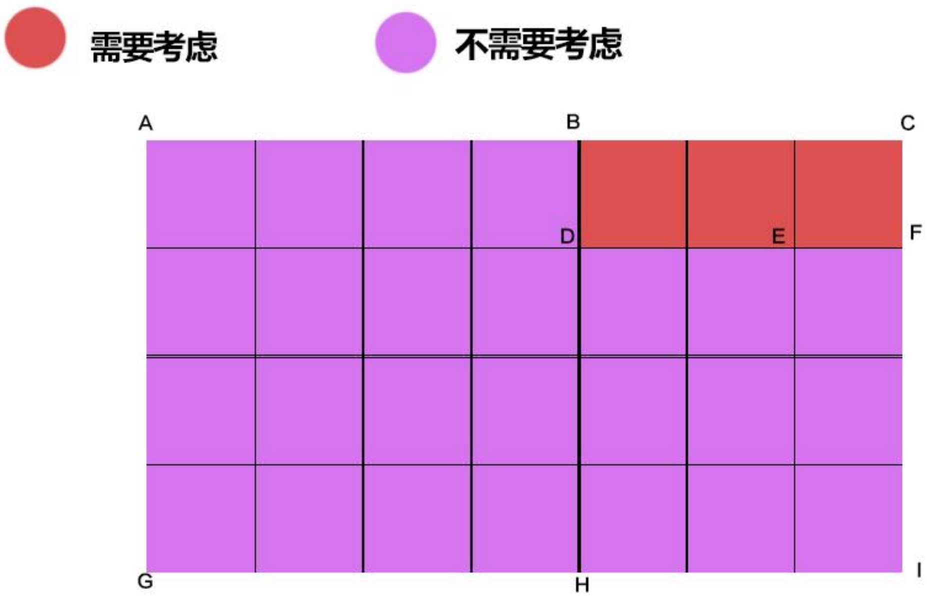
计算方法：用小边对大边进行除余，可以整除的部分即为若干个小边组成的正方形之和，故一定会被 gcd(a,b)整除，可以忽略，此时只考虑剩下的部分，之前的小边此时会变为剩余部分的大边，余数会变成剩余部分的小边，以此类推，当剩余部分的大边可以被小边整除时，小边即为最大公因数。



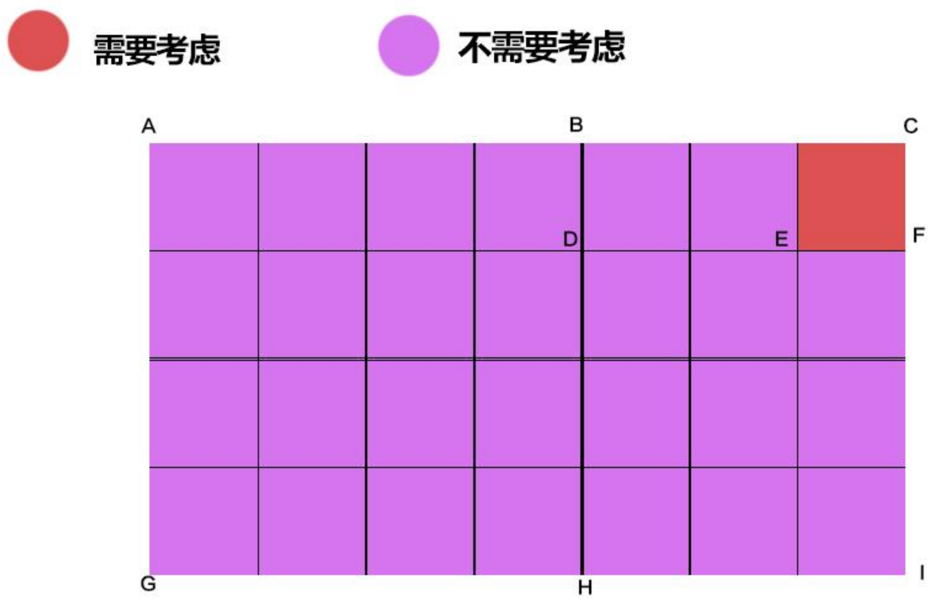
未被处理



1、　除余后 BH 为大边，BC 为小边



2、　再次除余后，大边为 BC，小边为 CF



3、 CF 可以整除 BC，因此，最大公因数为 CF

完整代码:

```cpp
#include<bits/stdc++.h>
using namespace std;

int gcd(int a,int b) {
    //让a变成大边，b变成小边
    if(a<b) {
        int temp=a;
        a=b;
        b=temp;
    }
    //当a可以被整除时，b就是最大公因数
    while(a%b!=0) {
        int temp=a%b;
        a=b;
        b=temp;
    }
    return b;
}


long long lcm(int a,int b) {
    long long ji=(long long)a*(long long)b;
    return ji/gcd(a,b);
}

int main( )
{
    int n;
    cin>>n;
    long long ans=1;
    for(int i=0;i<n;i++) {
        int temp;
        cin>>temp;
        ans=lcm(ans,temp);
    }
    cout<<ans%998244353;
    return 0;
}
```

缺点：当样本数量极多，样本大小极大时，long long 对未除模答案来说位数太小了，容易溢出导致答案错误。

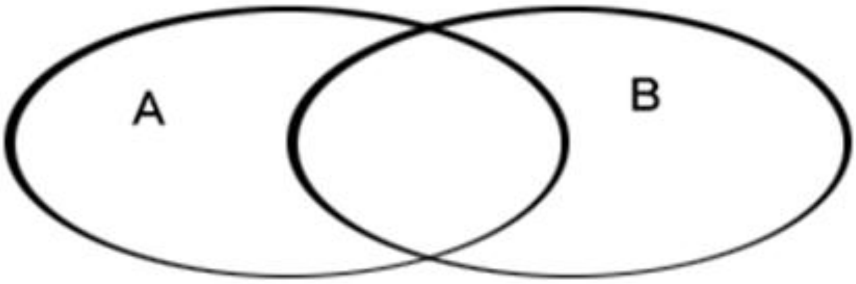
分解因式法：对于所有数来说，自身都可以被分解为若干质数的若干次幂之积

Eg. 8=2^3,48=2^4 * 3^1

对于两个数，最小公倍数即为二者中不同的底数*pow(相同的底数，相同的底数中最高次幂)

为了方便理解，我将引用一个例子 lcm(48,56)

48=2^4 * 3　　　　56=2^3 * 7



1.A∩B=2^3

2.A-A∩B=2*3

3.B-A∩B=7

lcm(48,56)=1.*2.*3.=2^3*2*3*7=2^4*3*7

由此方式，只需遍历出所有样本因式分解后的结果，将底数和指数都记录下来，当底数重复时将指数变为二者中较大的，最后将所有底数的指数次幂相乘即可

完整代码:

```cpp
#include<bits/stdc++.h>
using namespace std;

int main( )
{
    int n;
    cin>>n;
    unordered_map<int,int> di;//键值对存放底数和指数
    for(int i=0;i<n;i++) {
        int x;//输入样本
        cin>>x;
        for(int j=2;j*j<=x;j++) {//当j*j>x时，就说明已经没有可以因式分解出来的数了，剩下的x一定为1或质数
            if(x%j!=0) continue;
            int cnt=0;//记录指数
            while(x%j==0) {
                x/=j;
                cnt++;
            }
            di[j]=max(di[j],cnt);
        }
        if(x>1) di[x]=max(di[x],1);
    }
    int ans=1;
    for(auto p:di) ans=1ll*ans*(int)pow(p.first,p.second)%998244353;//1ll代表long long 形式的1，避免当ans过小时被隐式转换为int类型，导致溢出
    cout<<ans;
    return 0;
}
```

优点：因为在乘法过程中取模对结果没有影响，从而只需要 int 类型即可