数论

整除

如果 k_{1},k_{2} 互质，则 $k_{1}+k_{2}$ 与 $k_{1}×k_{2}$ 互质。

在自然数集中，小于 n 的质数约有 ${\frac {n} {ln(n)}}$ 个。

切比雪夫定理

$1.$ 对整数 $n□3$ □则至少存在一个质数 p □符合 $n□p□2n-2$ □

$2.$ 对任意自然数 $n > 6$ □ 至少存在一个 $4k + 1$ 型和一个 $4k + 3$ 型素数 p 使得 $n < p < 2n$ □

$3.$ 对任意自然数 k □ 存在自然数 N □ 对任意自然数 $n > N$ 至少存在 k 个素数 p 使得 $n < p < 2n$ □

$Miller-Rabin$

$Miller-Rabin$ 的复杂度是 $O(klogn)$ □其中 k 是测试次数。

质数筛法

埃氏筛

思想： 从小到大枚举分析每一个数，然后同时把当前这个数的所有（比自己大的）倍数记为合数，那么运行结束的时候没有被标记的数就是素数了。

```
int v[N];
void primes(int n) {
    memset(v, 0, sizeof v);
    for(int i = 2;i <= n; ++ i){
        if(v[i])continue;
        for(int j = i;j <= n / i; ++ j) v[i * j] = 1;
    }
}
```

时间复杂度□ $O(nlog_{10}log_{10}n)≈O(n)$ □所以它的时间复杂度其实是劣于线性筛的。这里补充自然数以及合数的和都是 $O(log_{10}n)$ □质数为 $O(log_{10}log_{10}n)$ □

虽然其时间复杂度比较劣，但这种思想是很值得学习的。如果需要筛一个 $[L,R]$ 的区间内的素数，我们需要先看 $sqrt(R)$ 的范围，然后预处理出这个范围内的素数。然后从小到大枚举素数，找到不小于 L 的最小 p 的倍数，且不能是 p 本身，然后按照这个筛法打标记，复杂度是 $O(R-L+{\sqrt {R}})$ □

代码如下：

```
memset(st, 0, sizeof st);
    for (int i = 0; i < cnt; i ++ ) {
        LL p = primes[i];//先筛一遍
        for (LL j = max(p * 2, (l + p - 1) / p * p); j <= r; j += p)
            st[j - l] = true;
    }
```

线性筛（欧拉筛）

扫到一个数字 i 时，如果没有标记过，则为质数。

不然 i 为合数，考虑将质数数组中从小到大开始给 $i×prime[j]$ 打标记，直到 $i%prime[j]==0$ ，这时我们找到了 i 的最小质因子 $prime[j]$ ，在此之前的质因子全都比 i 的最小质因子要小，所以打标记的数字，都是因为枚举到了这个数字的最小质因子才打的，之后的数字，因为都比最小质因子要大，所以不打标记，直接 $break$ ，所以每个数字因为只有一个最小质因子，所以只枚举了一次，复杂度为 $O(n)$ ，并且我们在筛的同时，也拿到了 $1~n$ 每个数字的最小质因子。

```
const int N = 10005;
int n, primes[N], cnt,min_prime[N];
bool vis[N];
inline void get_prime() {
    for(register int i = 2; i <= n; i ++) {
        if(!vis[i]) primes[ ++ cnt] = i;
        for(register int j = 1; j <= cnt && i * primes[j] <= n; ++ j) {
            vis[i * primes[j]] = 1;
            if(i % primes[j] == 0) {
                min_prime[i]=prime[j];//最小质因子
                break;
            }
        }
    }
}
```

反素数

如果 n 是 $1...n$ 中正约数个数最多的数，且唯一，也就是约数最多且最小，那么 n 就是反素数。

若 $N≤2^{31}$ ，$1...N$ 中任何数的不同质因子都不会超过 10 个且所有质因子的质数都不会超过 30 。因为光 2 乘到 31 这个数都比 N 大。所以反素数都可以表示为 $2^{c_{1}}×3^{c_{2}}×5^{c_{3}}×7^{c_{4}}×11^{c_{5}}×13^{c_{6}}×17^{c_{7}}×19^{c_{8}}×23^{c_{9}}×29^{c_{10}}$ ，其中 c 数组递减。

所以我们可以直接 dfs 找到前十个质数

例 1

题目

给定一个正整数 n ，输出最小的整数，满足这个整数有 n 个因子，即求因子数一定的最小反素数。

题解

按上面的剪剪枝，乱写就行了。

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
typedef unsigned long long ull;

int num[12]={2,3,5,7,11,13,17,19,23,29};
int cnt[12];
int n;
ull ans=9e18;
void dfs(int now,ull q,int pos) {
    if(now==n) {
        if(q<ans) ans=q;
        return;
    }
    if(pos>=10) return;
    ull tmp=q;
    for(int i=1;(i+1)*now<=n;i++) {
        if(pos!=0) {
            if(i>cnt[pos-1]) break;
        }
        tmp*=num[pos];
        if (tmp>2e18) break;
        if(n/now%(i+1)!=0) continue;
        cnt[pos] = i;
        dfs(now*(i+1),tmp,pos+1);
    }
}

int main() {
    scanf("%d",&n);
    dfs(1,1,0);
    printf("%llu",ans);
    return 0;
}
```