Nhận xét:

* GCD(X, Y) % M !=z GCD(X % M, Y % M)
* Gọi , gọi
* Vì giá trị của rất lớn nên ta tìm cách biểu diễn giá trị của và . Giả sử có biểu diễn theo tích các thừa số nguyên tố như sau

trong đó là một thừa số nguyên tố lớn nhất của và là các số nguyên dương.

* Như vậy ta có
* Ta biểu diễn bằng cách lưu trữ lại số mũ trong phân tích thừa số nguyên tố. Ví dụ: .

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | … |
| 0 | 0 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

* Ta phân tích từng phần tử và cộng dồn các số mũ tương ứng với nhau.

exp: mảng lưu trữ các mũ trong phân tích thừa số nguyên tố

void Factorize(int n)

{

int a = 2, m = sqrt(n);

while (n > 1 && a <= m)

{

if (n % a == 0)

{

++exp[a];

n = n / a;

}

else ++a;

}

if (n > 1) ++exp[n];

}

* Đoạn chương trình trên không hiệu quả vì phải xét nhiều giá trị là hợp số. Để cải tiến ta chỉ xét là các số nguyên tố.

p: là mảng chứa các số nguyên tố, np là số phần tử của mảng p

void Factorize(int n)

{

int i = 0, m = sqrt(n);

while (n > 1 && i < np && p[i] <= m)

{

if (n % p[i] == 0)

{

++exp[p[i]];

n = n / p[i];

}

else ++i;

}

if (n > 1) ++exp[n];

}

* Đoạn chương trình trên vẫn chưa hiệu quả vì phải xét nhiều số nguyên tố nhưng không phải là ước nguyên tố. Ví dụ phân tích số

|  |  |
| --- | --- |
| n | a |
| 95 | d[95] |
| 19 | d[19] |
| 1 |  |

* Gọi là một thừa số nguyên tố của số nguyên . Quy ước là số nguyên tố thì . Ví dụ: hoặc .

void Factorize(int n)

{

while (n > 1)

{

p = d[n];

if (p == 0) p = n;

while (n % p == 0)

{

++exp[p];

n = n / p;

}

}

}

Độ phức tạp của hàm phân tích trên là