Euclidean Algorithm

우선 약수의 개념을 알아보자.

a가 b의 약수이다. → a = bk (k는 음이 아닌 정수)

이는 a|b라 작성할 수 있다. 예를 들어 0의 약수를 구한다면 이 때 값은 모든 양의 정수가 된다.

0 = a*k를 만족하고 k = 0 일 때, a는 모든 정수이면 되기 때문이다.

최대 공약수

a, b가 최대공약수 k를 가지고 있다면 다음과 같이 표현할 수 있다.

gcd(a,b) = k

a = k*a′,   b = k*b′

gcd(a′, b′) =1

마지막 식에서 gcd(a′, b′)이 1이 아니라면 a, b는 k보다 더 큰 값을 최대 공약수를 가지기 때문에 모순이 된다. 이러한 약수 성질을 활용하여 유클리드 호제법은 다음과 같이 정의가 된다.

$A = B * q + r$ $A > B,\ \ 0\ \ <= r < B$ 를 만족하면

$gcd(A, B) = gcd(B, r)$ 이 된다.

$proof$

$$gcd(A,B) = g \quad g\,! = 1$$
$$A = g * \alpha$$
$$B = g * \beta$$
$$gcd(\alpha,\beta) = 1$$
$$g * \alpha = g * \beta * q + r$$
$$r = g * (\alpha - \beta * q)$$
$(\alpha - \beta * q) = r'$이라 치환한다면

$gcd(B,r) = g, \qquad gcd(r',\beta) = 1$을 만족해야 한다.

마찬가지로 $gcd(r',\beta)\,! = 1$이라 한다면, 귀류법에 의하여 모순이 생기게 된다.

(A, B가 g의 배수로 최대공약수를 가지게 된다.)

따라서, $gcd(A,B) = gcd(B,r)$ 만족하게 된다. 코드는 다음과 같다.

```
int gcd(int a, int b){
    int n;

    while(b!=0){
     n = a%b;
     a =b;
     b = n;
    }
    return a;
}
```