Euclidean Algorithms

In mathematics, the Euclidean algorithm or Euclid's algorithm, is an efficient method for computing the greatest common divisor (GCD) of two integers (numbers), the largest number that divides them both without a remainder.

- Euclid - Laws of nature are just the mathematical thoughts of God.
- Ancient Greek mathematician Euclid in Alexandria, Ptolemaic Egypt c. 300 BC.
- Father of Geometry

EXAMPLE

Find the greatest common divisor of $30, 36,$ and 24.

The divisors of each number are given by

$$\begin{aligned} 30 &: 1, 2, 3, 5, 6, 10, 15, 30 \\ 36 &: 1, 2, 3, 4, 6, 9, 12, 18, 36 \\ 24 &: 1, 2, 4, 6, 12, 24 \end{aligned}$$

The largest number that appears on every list is 6, so this is the greatest common divisor:

$$\gcd(30, 36, 24) = 6. \square$$

How to Find the Greatest Common Divisor?

For a set of two positive integers (a, b) we use the below-given steps to find the greatest common divisor:

- **Step 1:** Write the divisors of positive integer "a".
- **Step 2:** Write the divisors of positive integer "b".
- **Step 3:** Enlist the common divisors of "a" and "b".
- **Step 4:** Now find the divisor which is the highest of both "a" and "b".

Example: Find the greatest common divisor of 13 and 48.

Solution: We will use the below steps to determine the greatest common divisor of (13, 48).

Divisors of 13 are 1, and 13.

Divisors of 48 are 1, 2, 3, 4, 6, 8, 12, 16, 24 and 48.

The common divisor of 13 and 48 is 1.

The greatest common divisor of 13 and 48 is 1.

Thus, GCD(13, 48) = 1.

Euclid's Algorithm for Greatest Common Divisor

As per Euclid's algorithm for the greatest common divisor, the GCD of two positive integers (a, b) can be calculated as:

- If a = 0, then GCD (a, b) = b as GCD (0, b) = b.
- If b = 0, then GCD (a, b) = a as GCD (a, 0) = a.
- If both a≠0 and b≠0, we write 'a' in quotient remainder form (a = b×q + r) where q is the quotient and r is the remainder, and a>b.
- Find the GCD (b, r) as GCD (b, r) = GCD (a, b)
- We repeat this process until we get the remainder as 0.

Example: Find the GCD of 12 and 10 using Euclid's Algorithm.

Solution: The GCD of 12 and 10 can be found using the below steps:

$a = 12$ and $b = 10$

$a \neq 0$ and $b \neq 0$

In quotient remainder form we can write $12 = 10 \times 1 + 2$

Thus, GCD (10, 2) is to be found, as GCD(12, 10) = GCD (10, 2)

Now, $a = 10$ and $b = 2$

$a \neq 0$ and $b \neq 0$

In quotient remainder form we can write $10 = 2 \times 5 + 0$

Thus, GCD (2,0) is to be found, as GCD(10, 2) = GCD (2, 0)

Now, $a = 2$ and $b = 0$

$a \neq 0$ and $b = 0$

Thus, GCD $(2,0) = 2$

GCD $(12, 10)$ = GCD $(10, 2)$ = GCD $(2, 0) = 2$

Thus, GCD of 12 and 10 is 2.

Euclid's algorithm is very useful to find GCD of larger numbers, as in this we can easily break down numbers into smaller numbers to find the greatest common divisor.

Example:

Find the GCD of 270 and 192

- A=270, B=192
- A ≠0
- B ≠0
- Use long division to find that 270/192 = 1 with a remainder of 78. We can write this as: 270 = 192 * 1 +78
- Find GCD(192,78), since GCD(270,192)=GCD(192,78)

A=192, B=78

- A ≠0
- B ≠0
- Use long division to find that 192/78 = 2 with a remainder of 36. We can write this as:
- 192 = 78 * 2 + 36
- Find GCD(78,36), since GCD(192,78)=GCD(78,36)

A=78, B=36

- A $\neq 0$
- B $\neq 0$
- Use long division to find that 78/36 = 2 with a remainder of 6. We can write this as:
- 78 = 36 * 2 + 6
- Find GCD(36,6), since GCD(78,36)=GCD(36,6)

A=36, B=6

- A ≠0
- B ≠0
- Use long division to find that 36/6 = 6 with a remainder of 0. We can write this as:
- 36 = 6 * 6 + 0
- Find GCD(6,0), since GCD(36,6)=GCD(6,0)

A=6, B=0

- A ≠0
- B =0, GCD(6,0)=6

So we have shown:

GCD(270,192) = GCD(192,78) = GCD(78,36) = GCD(36,6) = GCD(6,0) = 6

GCD(270,192) = 6

Understanding the Euclidean Algorithm

If we examine the Euclidean Algorithm we can see that it makes use of the following properties:

- GCD(A,0) = A
- GCD(0,B) = B
- **If A = B·Q + R and B≠0 then GCD(A,B) = GCD(B,R)** where Q is an integer, R is an integer between 0 and B-1

The first two properties let us find the GCD if either number is 0. The third property lets us take a larger, more difficult to solve problem, and **reduce it to a smaller, easier to solve problem.**

The Euclidean Algorithm makes use of these properties by rapidly reducing the problem into easier and easier problems, using the third property, until it is easily solved by using one of the first two properties.

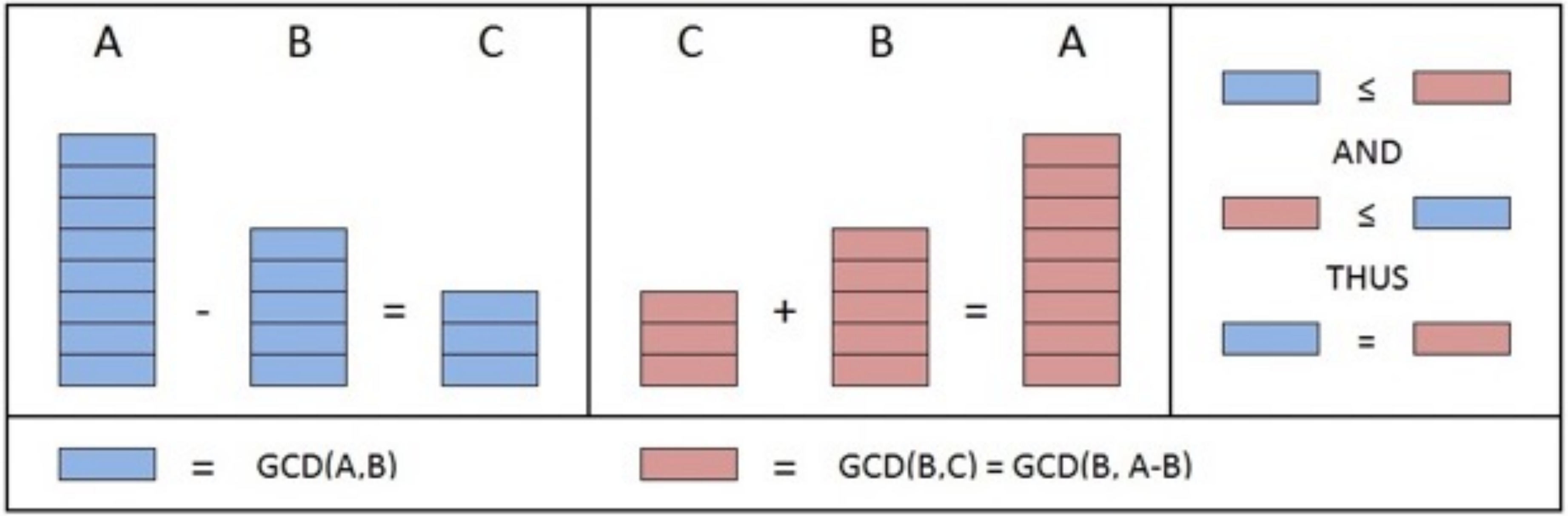
We can understand why these properties work by proving them.

We can prove that GCD(A,0)=A is as follows:

- The largest integer that can evenly divide A is A.
- All integers evenly divide 0, since for any integer, C, we can write $C \cdot 0 = 0$. So we can conclude that A must evenly divide 0.
- The greatest number that divides both A and 0 is A.

The proof for **GCD(0,B)=B** is similar. (Same proof, but we replace A with B).

To prove that **GCD(A,B)=GCD(B,R)** we first need to show that **GCD(A,B)=GCD(B,A-B)**.



Suppose we have three integers **A,B** and **C** such that **A-B=C**.

Proof that the GCD(A,B) evenly divides C

The GCD(A,B), by definition, evenly divides A. As a result, A must be some multiple of GCD(A,B). i.e. $X \cdot GCD(A,B)=A$ where X is some integer

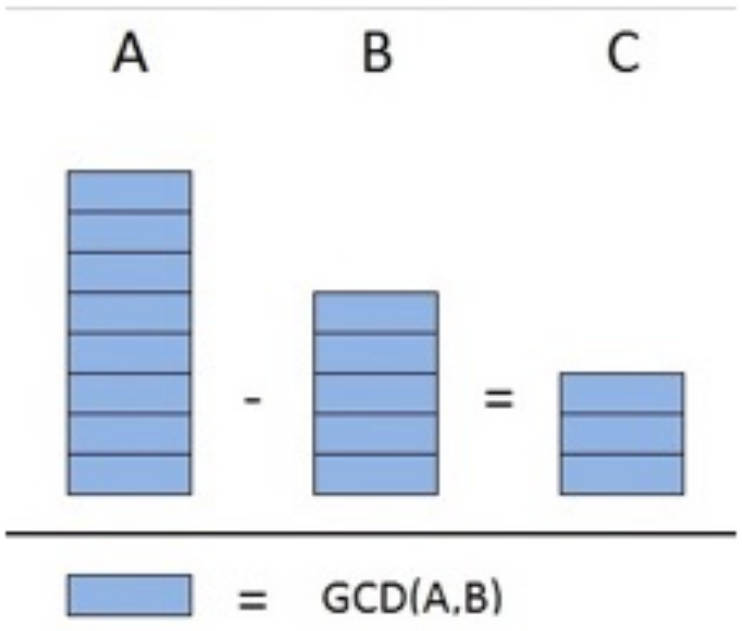
The GCD(A,B), by definition, evenly divides B. As a result, B must be some multiple of GCD(A,B). i.e. $Y \cdot GCD(A,B)=B$ where Y is some integer

$A-B=C$ gives us:

- $X \cdot GCD(A,B) - Y \cdot GCD(A,B) = C$
- $(X - Y) \cdot GCD(A,B) = C$

So we can see that GCD(A,B) evenly divides C.

An illustration of this proof is shown in the left portion of the figure below:



Proof that the GCD(B,C) evenly divides A

The GCD(B,C), by definition, evenly divides B. As a result, B must be some multiple of GCD(B,C). i.e. $M \cdot GCD(B,C)=B$ where M is some integer

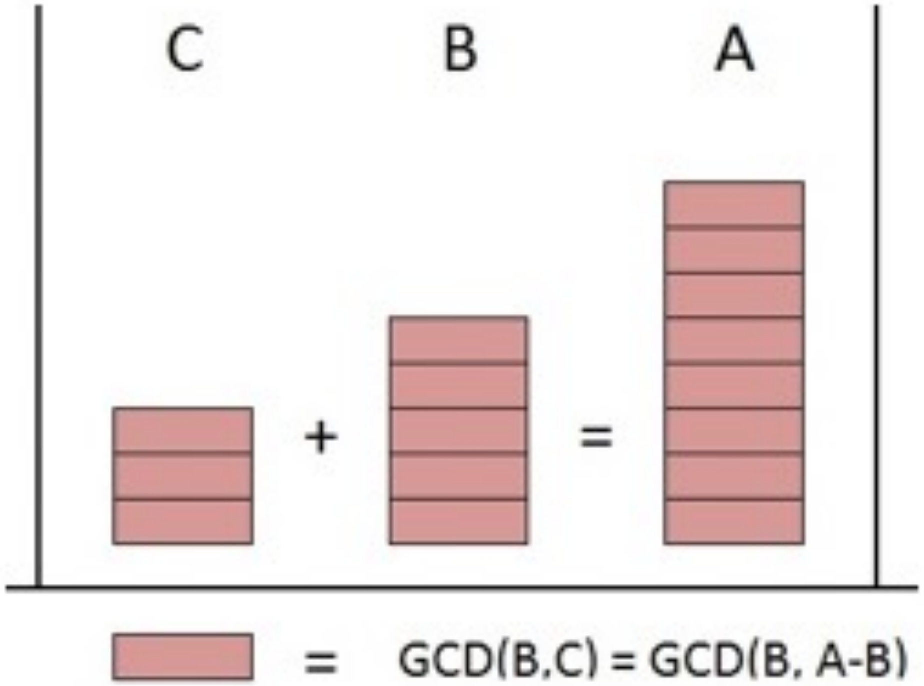
The GCD(B,C), by definition, evenly divides C. As a result, C must be some multiple of GCD(B,C). i.e. $N \cdot GCD(B,C)=C$ where N is some integer

$A-B=C$ gives us:

- $B+C=A$
- $M \cdot GCD(B,C) + N \cdot GCD(B,C) = A$
- $(M + N) \cdot GCD(B,C) = A$

So we can see that GCD(B,C) evenly divides A.

An illustration of this proof is shown in the figure below



Proof that GCD(A,B)=GCD(A,A-B)

- GCD(A,B) by definition, evenly divides B.
- We proved that GCD(A,B) evenly divides C.
- Since the GCD(A,B) divides both B and C evenly it is a common divisor of B and C.

GCD(A,B) must be less than or equal to, GCD(B,C), because GCD(B,C) is the "greatest" common divisor of B and C.

- GCD(B,C) by definition, evenly divides B.
- We proved that GCD(B,C) evenly divides A.
- Since the GCD(B,C) divides both A and B evenly it is a common divisor of A and B.

GCD(B,C) must be less than or equal to, GCD(A,B), because GCD(A,B) is the **"greatest"** common divisor of A and B.

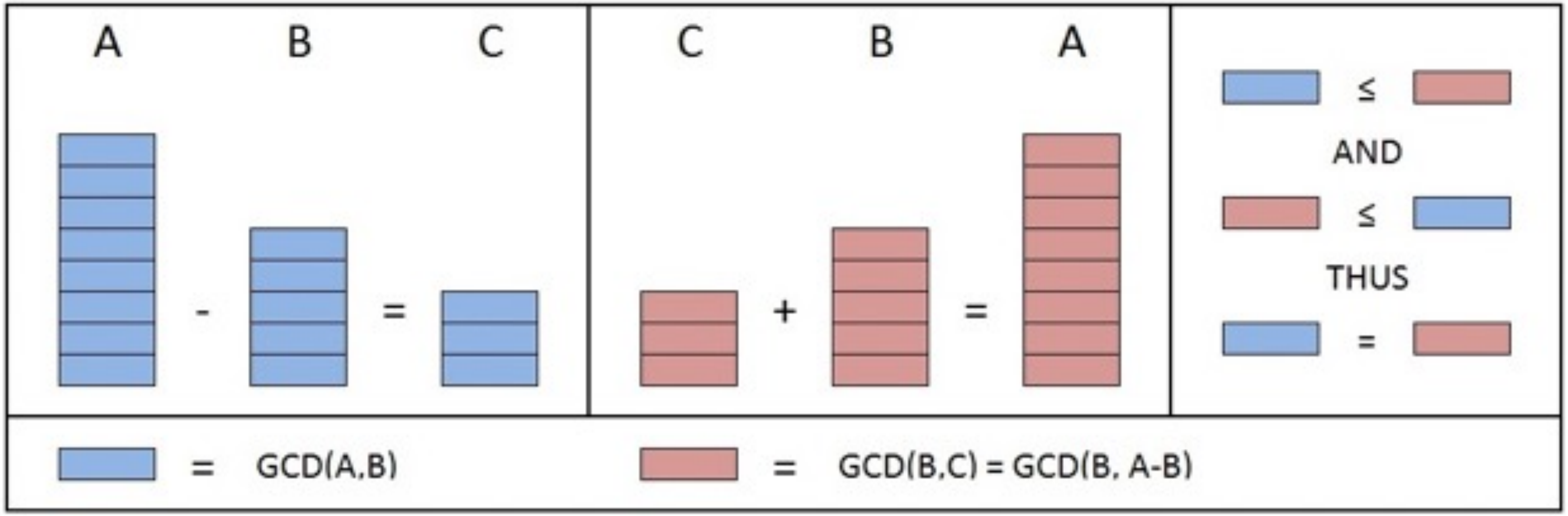
Given that $GCD(A,B) \leq GCD(B,C)$ and $GCD(B,C) \leq GCD(A,B)$ we can conclude that:

$$GCD(A,B)=GCD(B,C)$$

Which is equivalent to:

$$GCD(A,B)=GCD(B,A-B)$$

An illustration of this proof is shown in the right portion of the figure below.



Proof that GCD(A,B) = GCD(B,R)

We proved that GCD(A,B)=GCD(B,A-B)

The order of the terms does not matter so we can say GCD(A,B)=GCD(A-B,B)

We can repeatedly apply GCD(A,B)=GCD(A-B,B) to obtain:

GCD(A,B)=GCD(A-B,B)=GCD(A-2B,B)=GCD(A-3B,B)=...=GCD(A-Q·B,B)

But A= B·Q + R so A-Q·B=R

Thus **GCD(A,B)=GCD(R,B)**

The order of terms does not matter, thus:

GCD(A,B)=GCD(B,R)

```java
public class Euclid {

    // recursive implementation
    public static int gcd(int p, int q) {
        if (q == 0) return p;
        else return gcd(q, p % q);

    }

    // non-recursive implementation
    public static int gcd2(int p, int q) {
        while (q != 0) {
            int temp = q;
            q = p % q;
            p = temp;
        }
        return p;
    }
```

```java
    // main method
    public static void main(String[] args) {
        int p = Integer.parseInt(args[0]);
        int q = Integer.parseInt(args[1]);
        int d  = gcd(p, q); //resursion
        int d2 = gcd2(p, q); //while loop
        System.out.println("gcd(" + p + ", " + q + ") = " + d);
        System.out.println("gcd(" + p + ", " + q + ") = " + d2);
    }
}
```

```java
public class Euclid {

    // recursive implementation
    public static int gcd(int p, int q) {
        if (q == 0) return p;
        else return gcd(q, p % q);

    }

    // non-recursive implementation
    public static int gcd2(int p, int q) {
        while (q != 0) {
            int temp = q;
            q = p % q;
            p = temp;
        }
        return p;
    }

    // main method
    public static void main(String[] args) {
        int p = Integer.parseInt(args[0]);
        int q = Integer.parseInt(args[1]);
        int d  = gcd(p, q); //resursion
        int d2 = gcd2(p, q); //while loop
        System.out.println("gcd(" + p + ", " + q + ") = " + d);
        System.out.println("gcd(" + p + ", " + q + ") = " + d2);
    }
}
```