Math Explorations

Saviour Elikplim Animdife

2025-05-15

Contents

Preface **5**

1 GCD and LCM 1 **7**

2 GCD and LCM 2 **9**

3 Factorial **13**

4 Proving Heron's Formula **15**

Preface

Here, you will encounter my findings on my journey through the world of mathematics. I hope you find the material rewarding and worth exploring. Cheers!

Download the PDF version

Chapter 1

GCD and LCM 1

The GCD or greatest common divisor of two natural numbers (positive whole numbers) a and b is the largest number that can divide the two numbers without a remainder. For example, if we take 10 and 20 as our two natural numbers, 5 divides both without a remainder; $\frac{10}{5} = 2$ and $\frac{20}{5} = 4$. However, 5 is not the largest number that can divide the two numbers without a remainder. The largest number that can do this is 10 itself.

To learn how to use the Euclidean algorithm to work out GCDs by hand, refer to this site: https://sites.math.rutgers.edu/~greenfie/gs2004/euclid.html.

As an example, say we want to find the GCD of 90 and 35, then:

$$\begin{aligned} \gcd(90, 35) &= \gcd(35, 20) \\ &= \gcd(20, 15) \\ &= \gcd(15, 5) \\ &= \gcd(5, 0) \\ \text{Thus, } \gcd(90, 35) &= 5 \end{aligned}$$

The LCM is similar but as its name lowest common multiple suggests, the LCM of two natural numbers a and b is the smallest number c that is a multiple of a and is also a multiple of b. As an example, if we take two numbers 8 and 4, 16 is a multiple of both numbers but is not the smallest number that is a multiple of both numbers; the smallest number that is is 8 itself. Thus, the LCM$(8, 4)$ is 8.

The example below shows how to find the LCM of two natural numbers 500 and 75.

First, find the GCD of 500 and 75 using the Euclidean algorithm and then use the LCM formula to calculate the LCM based on the GCD and the absolute multiple of the two numbers.

$$\begin{aligned}
\gcd(500, 75) &= \gcd(75, 50) \\
&= \gcd(50, 25) \\
&= \gcd(25, 0) \\
\text{So, } \gcd(500, 75) &= 25
\end{aligned}$$

$$\begin{aligned}
\text{lcm}(500, 75) &= \frac{|500 \times 75|}{\gcd(500, 75)} \\
&= \frac{|37, 500|}{25} \\
&= \frac{37, 500}{25} \\
&= 1, 500 \\
\text{So, } \text{lcm}(500, 75) &= 1, 500
\end{aligned}$$

The Python code below demonstrates how to obtain the GCD and LCM of two natural numbers programmatically.

```
# Find GCD(a, b) using the Euclidean algorithm
def gcd(a, b):
    smaller = min(a, b)
    bigger = max(a, b)

    while (smaller != 0):
        next_bigger = smaller
        smaller = bigger % smaller
        bigger = next_bigger

    return bigger

# Find LCM(a, b)
def lcm(a, b):
    return int(abs(a * b) / gcd(a, b))

# Test LCM and GCD
print("LCM(500, 75): ", lcm(500, 75))
```

Parting words: Hopefully, you now understand how the LCM and GCD of two natural numbers are calculated or computed both by hand and programmatically.

Chapter 2

GCD and LCM 2

Previously, we looked at how to compute the GCD and LCM of two natural numbers both by hand and programmatically. However, what if we are interested in determining the GCD and LCM of four, five or any number of natural numbers? How do we do this? The answer is simple: break down the problem into manageable pieces which you then work through. So, for four numbers, we would find the GCD of the first two numbers, take that answer and find the GCD of the answer and the third number and finally find the GCD of the resulting answer and the last number. The same applies to the LCM. Finding the GCD and the LCM of more than two natural numbers is thus an iterative process that involves repeated use of the GCD and LCM functions for two natural numbers.

As an example, let us find the GCD of the numbers 500, 8, 50 and 70.

$$\begin{aligned} \gcd(500, 8) &= \gcd(8, 4) \\ &= \gcd(4, 0) \\ &= 4 \end{aligned}$$

$$\begin{aligned} \gcd(4, 50) &= \gcd(50, 4) \\ &= \gcd(4, 2) \\ &= \gcd(2, 0) \\ &= 2 \end{aligned}$$

$$\begin{aligned} \gcd(2, 70) &= \gcd(70, 2) \\ &= \gcd(2, 0) \\ &= 2 \end{aligned}$$

$$\text{So, the GCD of 500, 8, 50 and 70} = 2$$

Also, let's find the LCM of the four numbers 500, 8, 50 and 70.

$$\begin{aligned}\text{lcm}(500, 8) &= \frac{|500 \times 8|}{\text{gcd}(500, 8)} \\ &= \frac{4{,}000}{4} \\ &= 1{,}000\end{aligned}$$

$$\begin{aligned}\text{lcm}(1{,}000, 50) &= \frac{|1{,}000 \times 50|}{\text{gcd}(1{,}000, 50)} \\ &= \frac{1{,}000 \times 50}{50} \\ &= 1{,}000\end{aligned}$$

$$\begin{aligned}\text{lcm}(1{,}000, 70) &= \frac{|1{,}000 \times 70|}{\text{gcd}(1{,}000, 70)} \\ \text{But: } \text{gcd}(1{,}000, 70) &= \text{gcd}(70, 20) \\ &= \text{gcd}(20, 10) \\ &= \text{gcd}(10, 0) \\ &= 10 \\ \text{Thus, } \text{lcm}(1{,}000, 70) &= \frac{1{,}000 \times 70}{10} \\ &= 100 \times 70 \\ &= 7{,}000\end{aligned}$$

$$\text{Hence, the LCM of 500, 8, 50 and 70} = 7{,}000$$

The following Python code demonstrates how to programmatically determine the GCD and LCM of more than two numbers. The Python code relies on the function definitions for the gcd() and lcm() funtions from the previous GCD and LCM post.

```
# Assuming the gcd and lcm functions have already being
# defined or imported

# Extended version of the GCD function
def gcd_ext(arr):
    # arr is a list of natural numbers
    ans = gcd(arr[0], arr[1])
    i = 2
    while i < len(arr):
        ans = gcd(ans, arr[i])
        i += 1
```

```
    return ans

# Extended version of the LCM function
def lcm_ext(arr):
    ans = lcm(arr[0], arr[1])
    i = 2
    while i < len(arr):
        ans = lcm(ans, arr[i])
        i += 1
    return ans

# Test the extended versions of the GCD and LCM functions
print("GCD([500, 8, 50, 70]):", gcd_ext([500, 8, 50, 70]))  # Ans: 2
print("LCM([500, 8, 50, 70]):", lcm_ext([500, 8, 50, 70]))  # Ans: 7000
```

Chapter 3

Factorial

The factorial of a natural number n, that is, $n!$ is that number multiplied by all the natural numbers less than that number. Thus, $4! = 4 \times 3 \times 2 \times 1 = 24$. By definition, $0! = 1$.

The Python code below demonstrates how to programmatically determine the factorial of a natural number both non-recursively and using recursion.

```
def fact(n):
    list_n = list(range(1, n + 1))
    product = 1
    for number in list_n:
        product *= number

    return product

# Test factorial function: fact
print("Factorial of 5: ", fact(5))  # Ans: 120

def fact_r(n):
    if n == 0:
        return 1
    else:
        return n * fact_r(n - 1)

# Test recursive factorial function: fact_r
print("Recursive factorial of 4: ", fact_r(4))  # Ans: 24
```

Chapter 4

Proving Heron's Formula

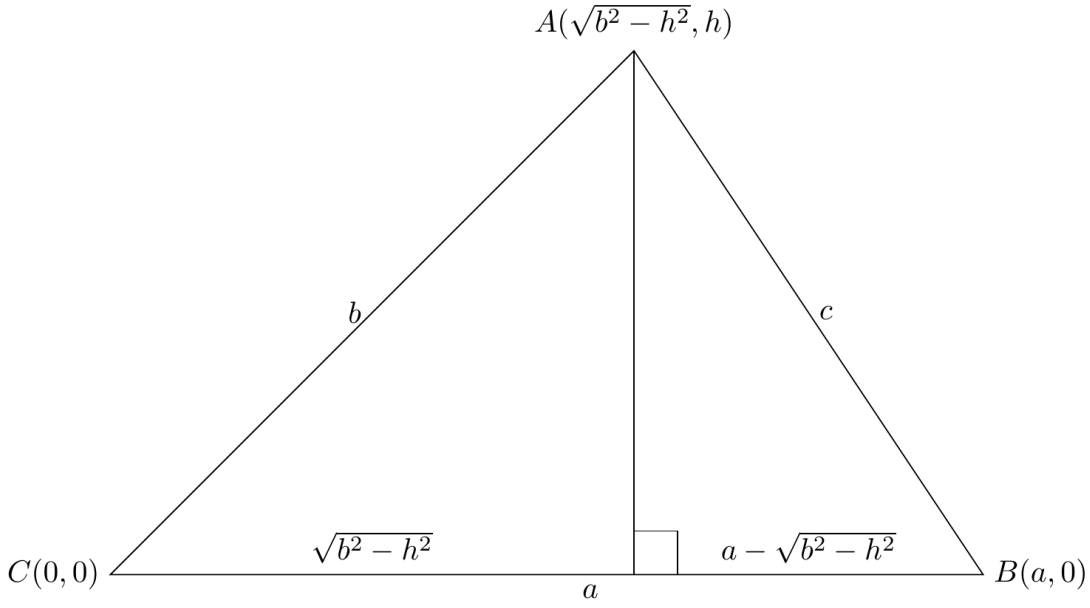

Figure 4.1: Labelled triangle with one vertex at the origin.

–>
Here is a simple proof of Heron's formula using Cartesian geometry. Here is a simple proof of Heron's formula using Cartesian geometry. Here is a simple proof of Heron's formula using Cartesian geometry. Here is a simple proof of Heron's formula using Cartesian geometry.

$$
\begin{aligned}
a - \sqrt{b^2 - h^2} &= \sqrt{c^2 - h^2} \\
(a - \sqrt{b^2 - h^2})^2 &= (\sqrt{c^2 - h^2})^2 \\
a^2 - 2a\sqrt{b^2 - h^2} + (b^2 - h^2) &= (c^2 - h^2) \\
a^2 + b^2 - h^2 - 2a\sqrt{b^2 - h^2} &= c^2 - h^2 \\
(a^2 + b^2 - c^2)^2 &= (2a\sqrt{b^2 - h^2})^2 \\
(a^2 + b^2 - c^2)(a^2 + b^2 - c^2) &= 4a^2(b^2 - h^2) \\
a^4 + b^4 + c^4 + 2a^2b^2 - 2a^2c^2 - 2b^2c^2 &= 4a^2b^2 - 4a^2h^2 \\
4a^2h^2 &= -a^4 - b^4 - c^4 - 2a^2b^2 + 4a^2b^2 + 2a^2c^2 + 2b^2c^2 \\
4a^2h^2 &= -a^4 - b^4 - c^4 + 2a^2b^2 + 2a^2c^2 + 2b^2c^2 \\
h &= \sqrt{\frac{-a^4 - b^4 - c^4 + 2a^2b^2 + 2a^2c^2 + 2b^2c^2}{4a^2}} \\
h &= \frac{1}{2a}\sqrt{-a^4 - b^4 - c^4 + 2a^2b^2 + 2a^2c^2 + 2b^2c^2} \\
\Delta &= \frac{1}{2} \times \text{base} \times \text{height, where base} = a \\
\Delta &= \frac{1}{2} \times a \times \frac{1}{2a}\sqrt{-a^4 - b^4 - c^4 + 2a^2b^2 + 2a^2c^2 + 2b^2c^2} \\
\Delta &= \frac{1}{4}\sqrt{-a^4 - b^4 - c^4 + 2a^2b^2 + 2a^2c^2 + 2b^2c^2} \\
\Delta &= \frac{1}{4}\sqrt{-a^4 - b^4 - c^4 + 2a^2b^2 + 2a^2c^2 - 2b^2c^2 + 4b^2c^2} \\
\Delta &= \frac{1}{4}\sqrt{4b^2c^2 - (a^4 + b^4 + c^4 - 2a^2b^2 - 2a^2c^2 + 2b^2c^2)} \\
\text{Let } \alpha &= a^4 + b^4 + c^4 - 2a^2b^2 - 2a^2c^2 + 2b^2c^2 \\
&= a^4 + b^4 + c^4 - a^2b^2 - a^2b^2 - a^2c^2 - a^2c^2 + b^2c^2 \\
&\qquad + b^2c^2 \\
&= a^4 - a^2b^2 - a^2c^2 + b^4 - a^2b^2 + b^2c^2 + c^4 - a^2c^2 \\
&\qquad + b^2c^2 \\
&= a^2(a^2 - b^2 - c^2) + b^2(b^2 - a^2 + c^2) + c^2(c^2 - a^2 + b^2) \\
&= a^2(a^2 - b^2 - c^2) + b^2(-a^2 + b^2 + c^2) \\
&\qquad + c^2(-a^2 + b^2 + c^2) \\
&= -a^2(-a^2 + b^2 + c^2) + b^2(-a^2 + b^2 + c^2) \\
&\qquad + c^2(-a^2 + b^2 + c^2) \\
&= (-a^2 + b^2 + c^2)(-a^2 + b^2 + c^2) \\
\alpha &= (-a^2 + b^2 + c^2)^2
\end{aligned}
$$

$$\text{Thus, } \Delta = \frac{1}{4}\sqrt{4b^2c^2 - (-a^2+b^2+c^2)^2}$$
$$\Delta = \frac{1}{4}\sqrt{(2bc)^2 - (-a^2+b^2+c^2)^2}$$
$$\Delta = \frac{1}{4}\sqrt{(2bc - (-a^2+b^2+c^2))(2bc + (-a^2+b^2+c^2))}$$
$$\Delta = \frac{1}{4}\sqrt{(2bc + a^2 - b^2 - c^2)(2bc - a^2 + b^2 + c^2)}$$
$$\text{Let } m = 2bc + a^2 - b^2 - c^2$$
$$m = a^2 - b^2 - c^2 + ab - ab + ac - ac + bc + bc$$
$$= (a^2 + ac - ab) + (-b^2 + bc + ab) + (-c^2 + bc - ac)$$
$$= a(a + c - b) - b(b - c - a) - c(c - b + a)$$
$$= a(a + c - b) + b(a + c - b) - c(a + c - b)$$
$$m = (a + b - c)(a + c - b)$$
$$\text{Let } n = 2bc - a^2 + b^2 + c^2$$
$$n = -a^2 + b^2 + c^2 + ab - ab + ac - ac + bc + bc$$
$$= (-a^2 + ac + ab) + (b^2 + bc - ab) + (c^2 + bc - ac)$$
$$= a(-a + c + b) + b(b + c - a) + c(c + b - a)$$
$$= a(b + c - a) + b(b + c - a) + c(b + c - a)$$
$$n = (a + b + c)(b + c - a)$$
$$\Delta = \frac{1}{4}\sqrt{mn}$$
$$= \sqrt{\frac{mn}{16}}$$
$$= \sqrt{\frac{(a+b-c)(a+c-b)(a+b+c)(b+c-a)}{16}}$$
$$= \sqrt{\frac{a+b+c}{2} \times \frac{b+c-a}{2} \times \frac{a+c-b}{2} \times \frac{a+b-c}{2}}$$
$$= \sqrt{\frac{a+b+c}{2} \times \frac{a+b+c-2a}{2} \times \frac{a+b+c-2b}{2} \times \frac{a+b+c-2c}{2}}$$
$$= \sqrt{\left(\frac{a+b+c}{2}\right) \times \left(\frac{a+b+c}{2} - a\right) \times \left(\frac{a+b+c}{2} - b\right) \times \left(\frac{a+b+c}{2} - c\right)}$$
$$\text{Let } S = \frac{a+b+c}{2},$$
$$\text{then } \Delta = \sqrt{S(S-a)(S-b)(S-c)}$$

Heron's formula proved!