

# Math Library in Python

---

## Math Library for Advanced Calculations

The `math` library is built-in, so you don't need to install anything to use it. You can import it simply by using `import math`.

### 1. Basic Functions

The `math` library provides several basic mathematical functions, similar to what you might find on a calculator.

```
import math

# Absolute value
abs_val = math.fabs(-10) # Output: 10.0

# Factorial
factorial_val = math.factorial(5) # Output: 120
```

### 2. Exponential and Logarithmic Functions

These functions are useful for growth calculations, interest calculations, and scientific computations.

- **Exponent** (`math.exp(x)`): Returns  $e$  raised to the power of  $x$ .
- **Logarithm:**
  - `math.log(x)`: Natural logarithm (base  $e$ ).
  - `math.log(x, base)`: Logarithm of  $x$  to a specified base.
  - `math.log10(x)`: Logarithm of  $x$  to base 10.
  - `math.log2(x)`: Logarithm of  $x$  to base 2.

```
exp_val = math.exp(2) # e^2
log_val = math.log(10) # ln(10)
log10_val = math.log10(100) # log_10(100)
```

### 3. Power and Root Functions

These functions are essential in algebra for handling exponents and square roots.

- **Power** (`math.pow(x, y)`): Returns  $x$  raised to the power of  $y$ .
- **Square Root** (`math.sqrt(x)`): Returns the square root of  $x$ .
- **Cube Root:** No direct function, but you can use `x ** (1/3)`.

```
power_val = math.pow(3, 4) # 3^4
sqrt_val = math.sqrt(16) # sqrt(16)
cube_root = 27 ** (1/3) # cube root of 27
```

## 4. Trigonometric Functions

The library includes all basic trigonometric functions, which are helpful in geometry, physics, and engineering.

- **Sine, Cosine, Tangent:**

- `math.sin(x)`: Sine of x (x is in radians).
- `math.cos(x)`: Cosine of x.
- `math.tan(x)`: Tangent of x.

- **Inverse Trigonometric Functions:**

- `math.asin(x)`: Inverse sine.
- `math.acos(x)`: Inverse cosine.
- `math.atan(x)`: Inverse tangent.

- **Conversion Functions:**

- `math.radians(degrees)`: Converts degrees to radians.
- `math.degrees(radians)`: Converts radians to degrees.

```
angle_rad = math.radians(90)    # Convert 90 degrees to radians
sin_val = math.sin(angle_rad)   # Sine of 90 degrees
```

## 5. Hyperbolic Functions

Hyperbolic functions are analogs of trigonometric functions for the hyperbola, used in calculus and complex analysis.

- **Hyperbolic Sine:** `math.sinh(x)`
- **Hyperbolic Cosine:** `math.cosh(x)`
- **Hyperbolic Tangent:** `math.tanh(x)`

```
sinh_val = math.sinh(1)
cosh_val = math.cosh(1)
tanh_val = math.tanh(1)
```

## 6. Angular Functions

The library also offers functions for converting between radians and degrees, which is essential for working with angles in various units.

- `math.degrees(x)`: Converts radians to degrees.
- `math.radians(x)`: Converts degrees to radians.

```
degrees = math.degrees(math.pi) # Output: 180
radians = math.radians(180)     # Output:  $\pi$  (approx 3.14159)
```

## 7. Special Constants

The `math` library provides access to several useful mathematical constants:

- **Pi** (`math.pi`): Ratio of a circle's circumference to its diameter (~3.14159).
- **Euler's Number** (`math.e`): Base of the natural logarithm (~2.71828).
- **Tau** (`math.tau`): Ratio of a circle's circumference to its radius (~6.28318).

```
print(math.pi)    # 3.14159
print(math.e)      # 2.71828
print(math.tau)    # 6.28318
```

## 8. Rounding and Precision Functions

Rounding functions are useful for rounding numbers to the nearest integer or truncating decimal values.

- **Ceiling** (`math.ceil(x)`): Rounds  $x$  up to the nearest integer.
- **Floor** (`math.floor(x)`): Rounds  $x$  down to the nearest integer.
- **Truncate** (`math.trunc(x)`): Truncates  $x$  to the integer part only.

```
ceil_val = math.ceil(4.3)    # Output: 5
floor_val = math.floor(4.7)  # Output: 4
trunc_val = math.trunc(4.9)  # Output: 4
```

## 9. Greatest Common Divisor and Least Common Multiple

- **GCD** (`math.gcd(x, y)`): Finds the greatest common divisor of  $x$  and  $y$ .
- **LCM** (`math.lcm(x, y)`): Finds the least common multiple of  $x$  and  $y$  (available in Python 3.9+).

```
gcd_val = math.gcd(8, 12)    # Output: 4
lcm_val = math.lcm(4, 6)     # Output: 12 (Python 3.9+)
```

## 10. Combinatorics

The `math` library also has functions for combinatorics, such as combinations and permutations, useful in probability and statistics.

- **Permutations** (`math.perm(n, k)`): Returns the number of ways to arrange  $k$  items from  $n$  items.
- **Combinations** (`math.comb(n, k)`): Returns the number of ways to choose  $k$  items from  $n$  items without repetition.

```
perm_val = math.perm(5, 3)    # 5P3
comb_val = math.comb(5, 3)    # 5C3
```

Summary Table

Function Type	Examples
Basic Functions	<code>math.fabs</code> , <code>math.factorial</code>
Exponential and Logarithmic	<code>math.exp</code> , <code>math.log</code> , <code>math.log10</code>
Power and Roots	<code>math.pow</code> , <code>math.sqrt</code>
Trigonometric Functions	<code>math.sin</code> , <code>math.cos</code> , <code>math.tan</code>
Hyperbolic Functions	<code>math.sinh</code> , <code>math.cosh</code> , <code>math.tanh</code>
Angular Conversion	<code>math.degrees</code> , <code>math.radians</code>
Special Constants	<code>math.pi</code> , <code>math.e</code> , <code>math.tau</code>
Rounding and Precision	<code>math.ceil</code> , <code>math.floor</code> , <code>math.trunc</code>
GCD and LCM	<code>math.gcd</code> , <code>math.lcm</code>
Combinatorics	<code>math.perm</code> , <code>math.comb</code>

The `math` library is a comprehensive toolset for performing a wide range of mathematical calculations, making Python versatile for both basic and advanced math applications. Let me know if you'd like further details or examples on any specific function!