**Chú ý: Tất cả những dòng code này phải cài thư viện math trong python. Cách cài: pip install math**

**1.** copysign(x, y)**:**

* **Chức năng:** Trả về một số có độ lớn tuyệt đối giống như x nhưng mang dấu của y.
* **Ví dụ:**

import math

print(math.copysign(5.0, -1.0)) # Kết quả: -5.0

print(math.copysign(-3.0, 2.0)) # Kết quả: 3.0

* **Ứng dụng:** Hữu ích khi bạn muốn đảm bảo một giá trị có dấu cụ thể mà không thay đổi độ lớn của nó.

**2.** frexp(x)**:**

* **Chức năng:** Phân tách số thực x thành một cặp (m, e) sao cho x = m \* 2\*\*e, với độ lớn tuyệt đối của m nằm trong khoảng [0.5, 1.0). m được gọi là *mantissa* (phần định trị) và e là *exponent* (số mũ).
* **Ví dụ:**

import math

print(math.frexp(10.0)) # Kết quả: (0.625, 4) vì 10.0 = 0.625 \* 2\*\*4

print(math.frexp(0.75)) # Kết quả: (0.75, 0) vì 0.75 = 0.75 \* 2\*\*0

print(math.frexp(0.0)) # Kết quả: (0.0, 0)

* **Ứng dụng:** Thường được sử dụng trong các tính toán mức thấp liên quan đến biểu diễn dấu phẩy động.

**3.** isclose(a, b, rel\_tol=1e-09, abs\_tol=0.0)**:**

* **Chức năng:** Kiểm tra xem hai giá trị dấu phẩy động a và b có "gần nhau" hay không. Nó sử dụng một công thức kết hợp cả sai số tương đối (rel\_tol) và sai số tuyệt đối (abs\_tol).
* **Công thức:** abs(a - b) <= max(rel\_tol \* max(abs(a), abs(b)), abs\_tol)
* **Ví dụ:**

Python

print(math.isclose(1.0, 1.000000001)) # Kết quả: True (sai số tương đối nhỏ)

print(math.isclose(1.0, 1.1, rel\_tol=0.01)) # Kết quả: False (sai số tương đối lớn hơn 1%)

print(math.isclose(0.0, 0.1, abs\_tol=0.2)) # Kết quả: True (sai số tuyệt đối nhỏ hơn 0.2)

* **Ứng dụng:** Rất hữu ích khi so sánh các kết quả tính toán dấu phẩy động, vì chúng thường có thể khác nhau một chút do sai số làm tròn.

**4.** isfinite(x)**:**

* **Chức năng:** Trả về True nếu x là một số hữu hạn (không phải vô cực hoặc NaN - Not a Number), và False ngược lại.
* **Ví dụ:**

import math

print(math.isfinite(10.5)) # Kết quả: True

print(math.isfinite(float('inf'))) # Kết quả: False

print(math.isfinite(float('nan'))) # Kết quả: False

* **Ứng dụng:** Quan trọng để kiểm tra tính hợp lệ của các giá trị số trong quá trình tính toán.

**5.** isinf(x)**:**

* **Chức năng:** Trả về True nếu x là vô cực dương hoặc vô cực âm, và False ngược lại.
* **Ví dụ:**

import math

print(math.isinf(float('inf'))) # Kết quả: True

print(math.isinf(-float('inf'))) # Kết quả: True

print(math.isinf(10.0)) # Kết quả: False

* **Ứng dụng:** Giúp xác định các trường hợp kết quả tính toán vượt quá giới hạn biểu diễn của số thực.

**6.** isnan(x)**:**

* **Chức năng:** Trả về True nếu x là NaN (Not a Number), và False ngược lại. NaN thường là kết quả của các phép toán không xác định (ví dụ: 0/0, căn bậc hai của số âm).
* **Ví dụ:**

import math

print(math.isnan(float('nan'))) # Kết quả: True

print(math.isnan(math.sqrt(-1))) # Kết quả: True (trên một số hệ thống)

print(math.isnan(10.0)) # Kết quả: False

* **Ứng dụng:** Cần thiết để xử lý các kết quả không hợp lệ trong tính toán số.

**7.** ldexp(x, i)**:**

* **Chức năng:** Trả về x \* 2\*\*i. Đây là phép toán ngược lại của frexp().
* **Ví dụ:**

import math

m, e = math.frexp(10.0)

print(math.ldexp(m, e)) # Kết quả: 10.0

print(math.ldexp(0.5, 3)) # Kết quả: 4.0 (0.5 \* 2\*\*3)

* **Ứng dụng:** Hữu ích khi bạn đã có phần định trị và số mũ và muốn tái tạo lại số ban đầu.

**8.** nextafter(x, y)**:**

* **Chức năng:** Trả về số dấu phẩy động tiếp theo sau x theo hướng của y. Nếu y lớn hơn x, nó trả về số nhỏ nhất lớn hơn x có thể biểu diễn được. Nếu y nhỏ hơn x, nó trả về số lớn nhất nhỏ hơn x có thể biểu diễn được.
* **Ví dụ:**

import math

print(math.nextafter(1.0, 2.0)) # Kết quả: Số nhỏ nhất lớn hơn 1.0

print(math.nextafter(1.0, 0.5)) # Kết quả: Số lớn nhất nhỏ hơn 1.0

print(math.nextafter(float('inf'), 1.0)) # Kết quả: Số dấu phẩy động lớn nhất hữu hạn

* **Ứng dụng:** Có thể hữu ích trong các thuật toán số phức tạp, đặc biệt khi cần kiểm soát độ chính xác ở mức độ rất nhỏ.

**9.** ulp(x)**:**

* **Chức năng:** Trả về "unit in the last place" (ulp) của một số dấu phẩy động x. Đây là khoảng cách giữa x và số dấu phẩy động có độ lớn tuyệt đối gần nhất khác với x.
* **Ví dụ:**

import math

print(math.ulp(1.0))

print(math.ulp(10.0))

(Giá trị cụ thể của ulp phụ thuộc vào biểu diễn dấu phẩy động của hệ thống.)

* **Ứng dụng:** Thường được sử dụng trong phân tích lỗi số và để hiểu rõ hơn về độ chính xác của các phép toán dấu phẩy động.