Documentation: Pow(x, n)

Description:

The function `myPow` calculates the value of `x` raised to the power of `n`, denoted as `x^n`. It efficiently computes the result using an iterative approach.

Signature:

class Solution:

    def myPow(self, x: float, n: int) -> float:

Parameters:

- `x`: A floating-point number representing the base.
- `n`: An integer representing the exponent.

Returns:

- A floating-point number, the result of `x` raised to the power `n`.

Examples:

1. **Example 1:**
    - Input: x = 2.00000, n = 10
    - **Output:** 1024.00000

2. **Example 2:**
    - **Input:** x = 2.10000, n = 3
    - **Output:** 9.26100

3. **Example 3:**
    - **Input:** x = 2.00000, n = -2
    - **Output:** 0.25000
    - **Explanation:** 2^-2 = 1/2^2 = 1/4 = 0.25

Constraints:

- 100.0 < x < 100.0
- 2^31 <= n <= 2^31-1
- n is an integer.
- Either `x` is not zero or `n > 0
- 10^4 <= x^n <= 10^4

Approach:

1. If `n` is 0, return 1.
2. If `n` is negative, invert `x` and make `n` positive.
3. Initialize `result` to 1.
4. Iterate through `n` in binary representation:
    - If the least significant bit of `n` is 1, multiply `result` by `x`.
    - Update `x` by squaring it.
    - Right shift `n` by 1 (equivalent to integer division by 2).
5. Return `result`.