Divide Two Integers Documentation

Problem Statement:

Given two integers `dividend` and `divisor`, divide `dividend` by `divisor` without using multiplication, division, and mod operator. The integer division should truncate toward zero, which means losing its fractional part.

Function Signature:

def divide(self, dividend: int, divisor: int) -> int:

Function to divide two integers without using multiplication, division, and mod operator.

:param dividend: An integer representing the dividend.

:param divisor: An integer representing the divisor.

:return: The quotient after dividing dividend by divisor.

Example:

Example usage:

solution = Solution()

print(solution.divide(10, 3)) # Output: 3

print(solution.divide(7, -3))   # Output: -2

Algorithm Explanation:

1. Handle edge cases:

- If dividend is 0, return 0.

- If divisor is 1, return dividend.

- If divisor is -1, check if dividend is within the 32-bit signed integer range and return appropriate value.

2. Determine the sign of the result by checking if the signs of dividend and divisor are different.

3. Convert both dividend and divisor to positive to simplify computation.

4. Initialize quotient to 0.

5. Loop until dividend is greater than or equal to divisor:

- Subtract divisor from dividend until dividend becomes smaller than divisor.

- Double the divisor and count to speed up the process.

6. If the result needs to be negative, negate it.

7. Ensure the result is within the 32-bit signed integer range.

Complexity Analysis:

- Time Complexity: O(log(dividend/divisor))

- The algorithm has a logarithmic time complexity because it repeatedly halves the divisor.

- Space Complexity: O(1)

- The algorithm uses only a constant amount of extra space for storing variables.