Problem 3749: Evaluate Valid Expressions

Problem Information

Difficulty: Hard
Acceptance Rate: 82.23%
Paid Only: Yes
Tags: Hash Table, Math, String, Divide and Conquer, Stack

Problem Description

You are given a string `expression` that represents a nested mathematical expression in a simplified form.

A **valid** expression is either an integer **literal** or follows the format `op(a,b)`, where:

* `op` is one of `"add"`, `"sub"`, `"mul"`, or `"div"`. * `a` and `b` are each valid expressions.

The **operations** are defined as follows:

* `add(a,b) = a + b` * `sub(a,b) = a - b` * `mul(a,b) = a * b` * `div(a,b) = a / b`

Return an integer representing the **result** after fully evaluating the expression.

Example 1:

Input: expression = "add(2,3)"

Output: 5

Explanation:

The operation `add(2,3)` means `2 + 3 = 5`.

Example 2:

Input: expression = "-42"

Output: -42

Explanation:

The expression is a single integer literal, so the result is -42.

Example 3:

Input: expression = "div(mul(4,sub(9,5)),add(1,1))"

Output: 8

Explanation:

* First, evaluate the inner expression: `sub(9,5) = 9 - 5 = 4` * Next, multiply the results: `mul(4,4) = 4 * 4 = 16` * Then, compute the addition on the right: `add(1,1) = 1 + 1 = 2` * Finally, divide the two main results: `div(16,2) = 16 / 2 = 8`

Therefore, the entire expression evaluates to 8.

Constraints:

* `1 <= expression.length <= 105` * `expression` is valid and consists of digits, commas, parentheses, the minus sign `'-'`, and the lowercase strings `"add"`, `"sub"`, `"mul"`, `"div"`. * All intermediate results fit within the range of a long integer. * All divisions result in integer values.

Code Snippets

C++:

```
class Solution {
public:
long long evaluateExpression(string expression) {

}
};
```

Java:

```
class Solution {
public long evaluateExpression(String expression) {

}
}
```

Python3:

```
class Solution:
def evaluateExpression(self, expression: str) -> int:
```