2235. Add Two Integers

Easy Math

Leetcode Link

Problem Description

The given LeetCode problem is a straightforward one. We are provided with two integers, denoted as `num1` and `num2`. Our task is to calculate the sum of these two integers and return that value. This is a basic arithmetic operation and does not require any special algorithms or data structures.

Intuition

The intuition behind the solution is based on elementary mathematics—specifically, the operation of addition, which is one of the first operations we learn in arithmetic. In programming, the addition of two numbers is achieved using the `+` operator. We simply take the two integer inputs and apply this operator to produce their sum. The implementation in Python directly translates to returning the result of `num1 + num2`. Given the simplicity of the problem, no additional optimization or complex logic is needed; we arrive at the solution in a single step of arithmetic addition.

Solution Approach

In the context of the provided solution, we are discussing an implementation that involves no complex algorithms, data structures, or design patterns. The solution approach consists of a single operation: arithmetic addition.

Here's a step-by-step walkthrough of the solution implementation:

1. The `Solution` class contains a method named `sum`, which accepts two parameters: `num1` and `num2`. These parameters are intended to be integers, as indicated by the type hints `int` in the method signature.
2. Inside the `sum` method, we perform the addition with the expression `num1 + num2`. This is the only operation performed in this method.
3. The result of this addition is then directly returned. In Python, the `+` operator between two integers gives us their sum, which is exactly what we need in this scenario.

The Python programming language abstracts away the details of how integers are added at a lower level (i.e., binary addition, carry handling), providing us with this simple and direct way to calculate the sum of two numbers.

This method is an application of the most fundamental of operations and requires no extra space or time complexities to consider. There is no conditional logic, looping, or data manipulation involved, making the solution concise and to the point.

Example Walkthrough

Let's go through a small example to illustrate the solution approach. Assume we are calling the `sum` method of the `Solution` class, and we want to find the sum of two numbers, `num1` which is 8 and `num2` which is 5.

1. We initialize our `Solution` class and create an instance of it. (Not explicitly shown in code but assumed as per object-oriented programming paradigms).
2. We call the `sum` method on our instance and pass the values `8` and `5` to `num1` and `num2` respectively:

```
result = Solution().sum(8, 5)
```

3. Inside the `sum` method, the procedure is straightforward. The method executes the addition of `num1` and `num2`; in this case, it will compute `8 + 5`.
4. The expression `num1 + num2` is then evaluated by the Python interpreter. Mathematically, it would be the addition of `8` with `5`, which totals to `13`.
5. The method then returns this result. In our example, `result` will hold the value `13`.
6. The returned value `13` is the result we expect for the sum of `8` and `5`. The function has successfully accomplished its purpose.

Through this example, we have demonstrated that the method does exactly what it is intended to do: it adds two integers and returns their sum with no additional complexity.

Python Solution

```python
class Solution:
    # This method calculates the sum of two numbers
    def sum(self, num1: int, num2: int) -> int:
        # Add the two numbers and return the result
        return num1 + num2

```

Java Solution

```java
// Class named Solution that contains the method sum.
class Solution {

    // Method named sum that takes two integers as parameters and returns their sum.
    // The method is public, meaning it can be accessed from outside the class,
    // and it is non-static, meaning it needs an instance of the Solution class to be invoked.
    public int sum(int num1, int num2) {
        // Returns the result of adding num1 and num2.
        return num1 + num2;
    }
}

```

C++ Solution

```cpp
// Define the Solution class
class Solution {
public:
    // Define the sum method that takes two integer parameters
    // and returns their sum
    int sum(int num1, int num2) {
        // Add num1 and num2 and return the result
        return num1 + num2;
    }
};

```

Typescript Solution

```typescript
/**
 * Calculates the sum of two numbers.
 * @param {number} num1 - The first number to add.
 * @param {number} num2 - The second number to add.
 * @returns {number} - The sum of num1 and num2.
 */
function sum(num1: number, num2: number): number {
    // Add the two numbers and return the result
    return num1 + num2;
}

```

Time and Space Complexity

The provided Python function simply calculates the sum of two integers. Let's analyze the time complexity and space complexity:

- **Time Complexity:** The operation is a direct arithmetic addition of two integers, which is a constant-time operation. Therefore, the time complexity of this function is `O(1)`.
- **Space Complexity:** The function uses a fixed amount of space; it stores two input variables and one output variable, with no additional space being used that is dependent on the input size. Thus, the space complexity is also `O(1)`.