plus one leetcode solution

Plus One LeetCode Solution is a common coding problem that many aspiring software developers face when preparing for technical interviews. This problem is straightforward yet challenges your understanding of arrays and number manipulation. In this article, we will delve into the problem statement, present various approaches to solve it, and provide a comprehensive explanation of the optimal solution, including a Python implementation. By the end of this article, you will have a clear understanding of the Plus One problem and how to tackle it efficiently.

Understanding the Problem Statement

The Plus One problem is defined as follows: You are given a non-empty array of digits representing a non-negative integer. The digits are stored such that the most significant digit is at the head of the array, and each element in the array contains a single digit. The goal is to increment the integer by one and return the resulting array of digits.

For example:
- Input: [1, 2, 3]
- Output: [1, 2, 4]

In this example, the integer represented by the array is 123. When we add one to it, we get 124, which is represented as [1, 2, 4].

However, there are edge cases to consider:
- Input: [9]
- Output: [1, 0]

In this case, adding one to 9 results in 10, which requires us to add an additional digit to the array.

Approaches to Solve the Plus One Problem

There are various approaches to solve the Plus One problem, each with its own pros and cons. Here are three common methods:

1. Simple Iterative Approach

This approach involves traversing the digits from the last index to the first, adding one to the last digit, and managing the carry if the digit becomes 10.

Steps:
- Start from the last digit.
- Add one to the last digit.
- If the result is less than 10, return the array.
- If the result is 10, set the last digit to 0 and carry over the one to the next digit.
- Continue this process until there are no more digits to process.

- If there’s still a carry after processing all digits, prepend 1 to the array.

2. Using Python's Built-in Functions

Another approach is to convert the array of digits into a single integer, perform the addition, and convert it back to an array. This method is less efficient as it involves type conversion, but it’s straightforward.

Steps:
- Convert the list of digits to a string and then to an integer.
- Add one to the integer.
- Convert back to a string and split it into individual digits, returning them as a list.

3. Optimized Approach Using List Manipulation

This is a more efficient approach that combines the strengths of both the iterative and built-in function methods. It minimizes the overhead of type conversion while handling carries elegantly.

Steps:
- Traverse the list from the last digit to the first.
- Increment the last digit and check for carry.
- If carry exists, continue adjusting the digits until there is no carry or until all digits are processed.
- If all digits are processed and still have a carry, insert a new digit at the front.

Optimal Solution Implementation

In this section, we will provide a Python implementation of the optimal solution, which utilizes the iterative approach while efficiently managing carries.

```python
def plusOne(digits):
n = len(digits)

for i in range(n - 1, -1, -1):
if digits[i] < 9:
digits[i] += 1
return digits
digits[i] = 0 Set current digit to 0 if it becomes 10

If we reach here, it means all digits were 9
return [1] + digits Prepend 1 to the list
```

Explanation of the Code

- We start by determining the length of the input array `digits`.
- We loop through the array in reverse order, checking each digit.
- If a digit is less than 9, we simply add one and return the updated list.
- If a digit is 9, we set it to 0 (handling the carry).
- After the loop, if we’ve set all digits to 0, we return a new list starting with 1 followed by the zeros.

Time and Space Complexity

When evaluating the efficiency of our solution, we should consider both time and space complexity.

Time Complexity

- The time complexity of our solution is O(n), where n is the number of digits in the input list. In the worst-case scenario, we might need to iterate through all digits once.

Space Complexity

- The space complexity is O(1) if we don’t count the output array. However, since we may need to return a new list (in cases where all digits are 9), the space complexity can be considered O(n) for the output.

Conclusion

In conclusion, the Plus One LeetCode solution presents a valuable exercise in array manipulation and carry handling. By understanding the problem statement and implementing various approaches, you can gain deeper insights into algorithm development. The iterative method provided is efficient and concise, making it a suitable choice for technical interviews. Mastering this problem not only enhances your coding skills but also prepares you for more complex challenges in coding interviews.

Frequently Asked Questions

What is the 'Plus One' problem on LeetCode?

The 'Plus One' problem on LeetCode requires you to take an array of digits representing a non-negative integer and increment the integer by one. The challenge is to return the resulting array of digits.

How do you approach solving the 'Plus One' problem?

To solve the 'Plus One' problem, you start from the least significant digit (the rightmost one) and add one to it. If it results in a value of 10, you set that digit to 0 and carry over 1 to the next digit. This process continues until there are no more digits to process or the carry is zero.

What is the time complexity of the optimal solution for 'Plus One'?

The time complexity of the optimal solution for the 'Plus One' problem is O(n), where n is the number of digits in the input array. This is because in the worst-case scenario, you may have to traverse the entire array.

Can the input array for 'Plus One' contain leading zeros?

No, the input array for the 'Plus One' problem is expected to represent a non-negative integer, which means it should not contain leading zeros, except in the case where the number itself is zero.

What edge cases should I consider when implementing the 'Plus One' solution?

You should consider edge cases such as an input array consisting of all 9s (e.g., [9, 9, 9]), which will require an increase in the size of the array. Also, consider the case when the input is [0], which should return [1].

What are common mistakes to avoid when solving 'Plus One' on LeetCode?

Common mistakes include not correctly handling the carry when incrementing digits, forgetting to account for the case where an additional digit is needed, and not returning the updated array properly.

How can I test my solution for the 'Plus One' problem?

You can test your solution by using a variety of test cases, including edge cases such as [9, 9, 9], [0], and [1, 2, 3]. Additionally, you can validate your output against expected results to ensure correctness.

Plus One Leetcode Solution

Find other PDF articles:

https://parent-v2.troomi.com/archive-ga-23-47/files?dataid=jhR75-4365&title=pocket-companion-for-physical-examination-and-health-assessment.pdf

Plus One Leetcode Solution

Back to Home: https://parent-v2.troomi.com