Problem 2816: Double a Number Represented as a Linked List

Problem Information

Difficulty: Medium
Acceptance Rate: 61.21%
Paid Only: No
Tags: Linked List, Math, Stack

Problem Description

You are given the `head` of a **non-empty** linked list representing a non- negative integer without leading zeroes.

Return _the_`head` _of the linked list after**doubling** it_.

Example 1:

![](https://assets.leetcode.com/uploads/2023/05/28/example.png)

Input: head = [1,8,9] **Output:** [3,7,8] **Explanation:** The figure above corresponds to the given linked list which represents the number 189. Hence, the returned linked list represents the number 189 * 2 = 378.

Example 2:

![](https://assets.leetcode.com/uploads/2023/05/28/example2.png)

Input: head = [9,9,9] **Output:** [1,9,9,8] **Explanation:** The figure above corresponds to the given linked list which represents the number 999. Hence, the returned linked list reprersents the number 999 * 2 = 1998.

Constraints:

* The number of nodes in the list is in the range `[1, 104]` * `0 <= Node.val <= 9` * The input is generated such that the list represents a number that does not have leading zeros, except the number `0` itself.

Code Snippets

C++:

```
/**
* Definition for singly-linked list.
* struct ListNode {
* int val;
* ListNode *next;
* ListNode() : val(0), next(nullptr) {}
* ListNode(int x) : val(x), next(nullptr) {}
* ListNode(int x, ListNode *next) : val(x), next(next) {}
* };
*/
class Solution {
public:
ListNode* doubleIt(ListNode* head) {

}
};
```

Java:

```
/**
* Definition for singly-linked list.
* public class ListNode {
* int val;
* ListNode next;
* ListNode() {}
* ListNode(int val) { this.val = val; }
* ListNode(int val, ListNode next) { this.val = val; this.next = next; }
* }
*/
class Solution {
public ListNode doubleIt(ListNode head) {

}
}
```

Python3:

```
# Definition for singly-linked list.
# class ListNode:
# def __init__(self, val=0, next=None):
# self.val = val
# self.next = next
class Solution:
def doubleIt(self, head: Optional[ListNode]) -> Optional[ListNode]:
```