

# Problem 306: Additive Number

## Problem Information

**Difficulty:** Medium

**Acceptance Rate:** 33.23%

**Paid Only:** No

**Tags:** String, Backtracking

## Problem Description

An **additive number** is a string whose digits can form an **additive sequence**.

A valid **additive sequence** should contain **at least** three numbers. Except for the first two numbers, each subsequent number in the sequence must be the sum of the preceding two.

Given a string containing only digits, return `true` if it is an **additive number** or `false` otherwise.

**Note:** Numbers in the additive sequence **cannot** have leading zeros, so sequence `1, 2, 03` or `1, 02, 3` is invalid.

**Example 1:**

**Input:** "112358" **Output:** true **Explanation:** The digits can form an additive sequence: 1, 1, 2, 3, 5, 8.  $1 + 1 = 2$ ,  $1 + 2 = 3$ ,  $2 + 3 = 5$ ,  $3 + 5 = 8$

**Example 2:**

**Input:** "199100199" **Output:** true **Explanation:** The additive sequence is: 1, 99, 100, 199.  $1 + 99 = 100$ ,  $99 + 100 = 199$

**Constraints:**

`1 <= num.length <= 35` `num` consists only of digits.

**\*\*Follow up:\*\*** How would you handle overflow for very large input integers?

## Code Snippets

### C++:

```
class Solution {  
public:  
    bool isAdditiveNumber(string num) {  
  
    }  
};
```

### Java:

```
class Solution {  
    public boolean isAdditiveNumber(String num) {  
  
    }  
}
```

### Python3:

```
class Solution:  
    def isAdditiveNumber(self, num: str) -> bool:
```