Problem 1800: Maximum Ascending Subarray Sum

Problem Information

Difficulty: Easy
Acceptance Rate: 66.31%
Paid Only: No
Tags: Array

Problem Description

Given an array of positive integers `nums`, return the **maximum** possible sum of an strictly increasing subarray in __`nums`.

A subarray is defined as a contiguous sequence of numbers in an array.

Example 1:

Input: nums = [10,20,30,5,10,50] **Output:** 65 **Explanation:**[5,10,50] is the ascending subarray with the maximum sum of 65.

Example 2:

Input: nums = [10,20,30,40,50] **Output:** 150 **Explanation:**[10,20,30,40,50] is the ascending subarray with the maximum sum of 150.

Example 3:

Input: nums = [12,17,15,13,10,11,12] **Output:** 33 **Explanation:**[10,11,12] is the ascending subarray with the maximum sum of 33.

Constraints:

* `1 <= nums.length <= 100` * `1 <= nums[i] <= 100`

Code Snippets

C++:

```
class Solution {
public:
int maxAscendingSum(vector<int>& nums) {

}
};
```

Java:

```
class Solution {
public int maxAscendingSum(int[] nums) {

}
}
```

Python3:

```
class Solution:
def maxAscendingSum(self, nums: List[int]) -> int:
```