1695

# 1695. Maximum Erasure Value

Solved

Medium | Topics | Companies | Hint

You are given an array of positive integers `nums` and want to erase a subarray containing **unique elements**. The **score** you get by erasing the subarray is equal to the **sum** of its elements.

Return *the **maximum score** you can get by erasing **exactly one** subarray.*

An array `b` is called to be a subarray of `a` if it forms a contiguous subsequence of `a`, that is, if it is equal to `a[l],a[l+1],...,a[r]` for some `(l,r)`.

**Example 1:**

```
Input: nums = [4,2,4,5,6]
Output: 17
Explanation: The optimal subarray here is [2,4,5,6].
```

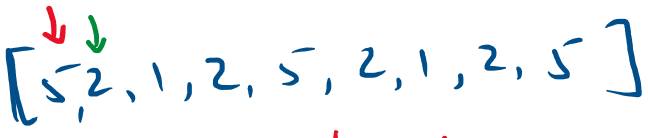
**Example 2:**

```
Input: nums = [5,2,1,2,5,2,1,2,5]
Output: 8
Explanation: The optimal subarray here is [5,2,1] or [1,2,5].
```

**Constraints:**

- `1 <= nums.length <= 10^5`
- `1 <= nums[i] <= 10^4`

Ans) Two pointer + store unique elements :




=> Iterate over each element

=> While the element is unique, add the value

Maintain a set to compare the curr Element with.

Return the maximum of all the Subarray sums.

```cpp
class Solution {
public:
    int maximumUniqueSubarray(vector<int>& nums) {
        // Pointer, Tracking curr subarr sum, result
        int ptr1 = 0, currentSum = 0, res = 0;
        // Hash set of Int to keep track of unique
elements
        unordered_set<int> uniqueElements;
        // Interating second pointer from prt1 to end of
array
        for(int ptr2 = 0; ptr2 < nums.size(); ptr2++){
            // If the number is not unique
            while (uniqueElements.find(nums[ptr2]) !=
uniqueElements.end()){
                // Clearing Hashset
                uniqueElements.erase(nums[ptr1]);
                currentSum -= nums[ptr1];
                // Trying Next subarray
                ptr1++;
            }
            // If number is unique add to currsum
            currentSum += nums[ptr2];
            uniqueElements.insert(nums[ptr2]);

            // Store max sum as the result
            res = max(res, currentSum);
        }
        // Retrun the result
        return res;
    }
};
```