05. CarryForward And Sub Arrays

14:08

Agenda :

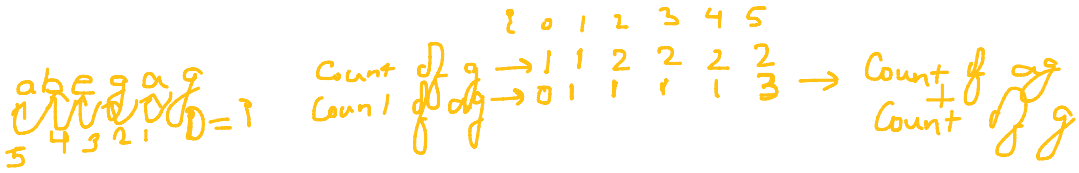
1. CarryForward And Problems
2. Subarrays and Problems

Carry Forward

Carry Forward is a technique in which we store the count of particular element and while iterating calculate the requirements to get the desired results.

Example :
Count of Pairs AG

"abegag" -> ans = 3.



Problems :

1. Given an integer array **A** containing **N** distinct integers, you have to find **all the leaders** in array **A**. An element is a leader if it is **strictly greater than** all the elements to its **right side.**
2. Given an array **A**, find the size of the smallest subarray such that it contains at least one occurrence of the maximum value of the array and at least one occurrence of the minimum value of the array.
3. Special Subsequence "AG"
4. Say you have an array, **A**, for which the **i**th element is the price of a given stock on day **i**.
   If you were only permitted to complete at most one transaction (ie, buy one and sell one share of the stock), design an algorithm to find the maximum profit.
   Return the **maximum** possible profit.
5. You are given a string **S**, and you have to find all the **amazing substrings** of **S**. An amazing Substring is one that starts with a **vowel** (a, e, i, o, u, A, E, I, O, U).
6. You are given an integer array **A**.
   Decide whether it is possible to divide the array into one or more subarrays of **even length** such that
   the **first** and **last** element of all subarrays will be **even**.
   Return "**YES**" if it is possible; otherwise, return "**NO**" (without quotes).
7. A wire connects **N** light bulbs.
   Each bulb has a switch associated with it; however, due to faulty wiring, a switch also changes the state of all the bulbs to the right of the current bulb.
   Given an initial state of all bulbs, find the **minimum number** of switches you have to press to turn on all the bulbs.
   You can press the same switch multiple times.
   Note: 0 represents the bulb is off and **1** represents the bulb is on.
8. You are given an integer array **A** of size **N**.
   You have to perform **B** operations. In one operation, you can remove either the leftmost or the rightmost element of the array **A**.
   Find and return the **maximum possible sum** of the **B elements** that were removed after the **B** operations.
   NOTE: Suppose **B = 3**, and array A contains 10 elements, then you can:
   - Remove 3 elements from front and 0 elements from the back, OR
   - Remove 2 elements from front and 1 element from the back, OR
   - Remove 1 element from front and 2 elements from the back, OR
   - Remove 0 elements from front and 3 elements from the back.

Introduction To Sub Arrays

A subarray is a contiguous part of an array. It is formed by selecting a range of elements from the array. A subarray can have one or more elements and must be a contiguous part of the original array.

Example

Consider the following array of integers:

4 1 2 3 -1 6 9 8 12

- 2, 3, -1, 6 is a subarray of length 4.
- 9 is a subarray of single element.
- 4, 1, 2, 3, -1, 6, 9, 8, 12 is a subarray consisting of all the array elements.
- 4, 12 is not a subarray because loop does not count as subarray.
- 1, 2, 6 is not a subarray because the array elements must be contiguous.
- 3, 2, 1, 4 is not a subarray because order of the elements in a subarray should be same as in the array.

A Subarray can be uniquely represented in following ways:

- By specifying the start and end index of the subarray.
- By specifying the start index and length of the subarray.

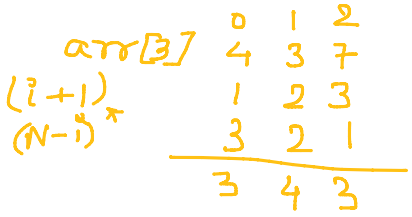
If we consider the same array from the above example,
4 1 2 3 -1 6 9 8 12
The subarray 2, 3, -1, 6 can be represented as

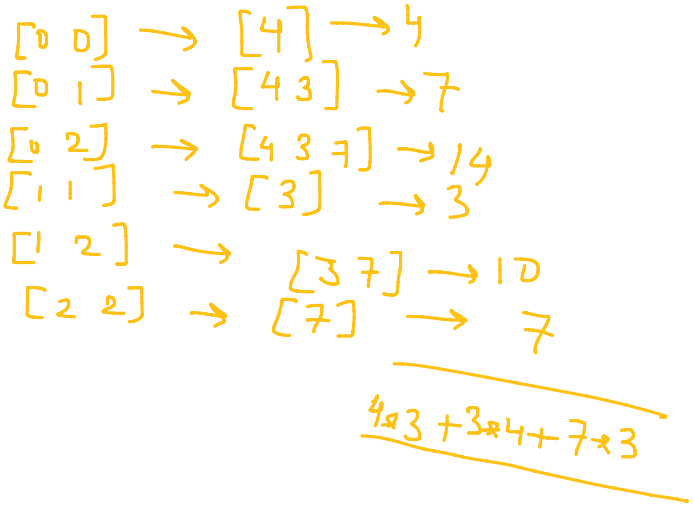
- the range of elements starting at index 2 and ending at index 5 (0-based indexing).
- the range of elements having length of 4 with start index as 2.

- **Total number of Sub arrays = n(n + 1) / 2, where n is the length of array.**

Contribution Technique

Consider a array [4, 3, 7] calculate sum of all subarrays.





Individual Contribution = 12 + 12 + 21
= 45

```
Int totalSum(int arr[]){
        Int n = arr.length;
        Int total = 0;
        For(i=0 -> N-1){
                Freq = (i+1)*(N-i);
                Con = arr[i]*freq;
                Total = total+con;
        }
}
```

Problems :

1. Given an array **A** of length **N**, return the subarray from **B** to **C**.

2. You are given an integer array **C** of size **A**. Now you need to find a subarray (contiguous elements) so that the sum of contiguous elements is maximum.
   But the sum must not exceed **B**.

3. You are given an integer array **A** of length **N.**
   You have to find the sum of all subarray sums of A.
   More formally, a subarray is defined as a contiguous part of an array which we can obtain by deleting zero or more elements from either end of the array.
   A subarray sum denotes the sum of all the elements of that subarray.

Note : Be careful of integer overflow issues while calculations. Use appropriate datatypes.

4. You are given an array **A** of **N** integers. Return a 2D array consisting of all the subarrays of the array.
   Note : The **order** of the subarrays in the resulting 2D array **does not matter**.

5. Given an array of integers **A**, a subarray of an array is said to be good if it fulfils any one of the criteria:
   1. Length of the subarray is be even, and the sum of all the elements of the subarray must be less than **B**
   2. Length of the subarray is be odd, and the sum of all the elements of the subarray must be greater than **B**.
   Your task is to find the count of good subarrays in A.

6. Given an array **A** of **N** non-negative numbers and a non-negative number **B,**
   you need to find the **number of subarrays in A with a sum less than B.**
   We may assume that there is no overflow.