## **Mohit and his Interesting Array**

**Problem Statement :**

For any Array A find the sum of elements present between the index L and index R inclusive. Array is called Interesting if the sum of the elements for Q queries is even or else it is not interesting.

**Note** :

Many of the people used Brute force to calculate the sum using a for loop but in such a case you would have faced TLE. Consider an array of 5 elements

Array 1 - (1 , 2, 3 , 4 , 5)

If there are 100 Queries defined with the index being 0 and 4 all the time your code would be calculating the sum again and again for 100 times which creates a lot of redundancy.

**Brute Force Approach** - (This Approach would have given TLE )

Algorithm :

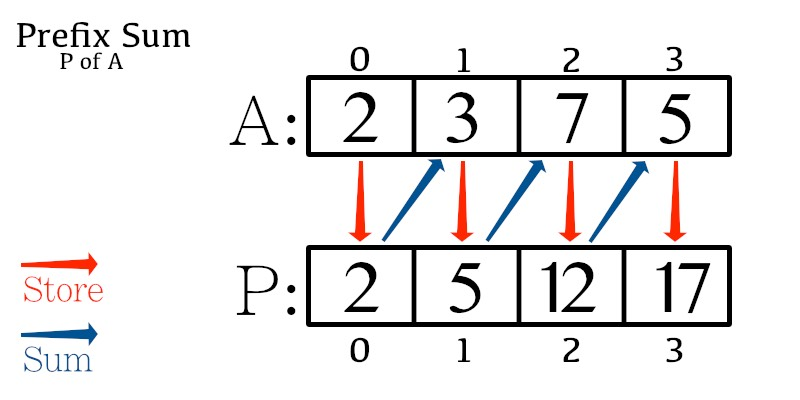
1. Take the Input
2. Declare a variable sum to store the sum
3. Loop for ‘Q’ Times // No of Queries given in the question
   1. Loop in the Array from Index L to Index R
   2. Store the sum in a variable
4. Once the Loop ends Check if sum is even or odd
   1. If sum is even print YES
   2. Else print NO

**Time Complexity Analysis - O()**

Consider an Array with Length ‘ N ’ and ‘Q’ Queries. If Q == N and the Indexes given are L = 0 and R = N-1 then the loop will run ‘N‘ times for ‘Queries and then internally for calculating the sum from index 0 to N-1 it will run for ‘N’ times. So in total it runs for times hence the time complexity is O**().**

**Efficient Approach -**

In order to resolve the TLE we use the concept of Prefix Sum Array



**Fig 1:** Prefix Sum Array Concept

To fill the prefix sum array, we run through index 1 to last and keep on adding the present element with previous value in prefix sum array.

Algorithm:

1. Take the input
2. Define a variable sum = 0 and PrefixSumArray of size ‘N’
3. Calculate Prefix Sum Array
   1. Loop from 1 to n-1
   2. Prefix[i] = Prefix[i -1] + Arr[i]
4. Loop till ‘Q’ // No of Queries
   1. Sum between index L and index R can be defined as
      1. Sum = Prefix[R] - Prefix[L-1]
   2. Add all the sum’s
5. Once the Loop ends Check if sum is even or odd
   1. If sum is even print YES
   2. Else print NO

Time Complexity Analysis - O(n)

In this case since operation **Sum = Prefix[R] - Prefix[L-1]** is of type O(1) we have reduced the time complexity to O(n)

Consider No of Queries to be ‘N’ but this time we only do O(1) task to actually calculate the **Sum = Prefix[R] - Prefix[L-1]** and due to which a lot of computation is reduced and our time complexity reduces to O(1)

**C++ Solution**

|  |
| --- |
| #include <cmath>  #include <cstdio>  #include <vector>  #include <iostream>  #include <algorithm>  using namespace std;  int main() {  int t;  cin>>t;  while(t-->0)  {  int n,q;  cin>>n>>q;  int arr[n];  int prefix[n];  int sum = 0;  for(int i=0;i<n;i++)  {  cin>>arr[i];  }  for(int i=1;i<n;i++)  {  prefix[i] = prefix[i-1] + arr[i];  }  for(int i=0;i<q;i++)  {  int l,r;  cin>>l>>r;  sum+= prefix[r] - prefix[l-1];  }  if(sum % 2==0)  cout<<"YES"<<endl;  else  cout<<"NO"<<endl;  }  return 0;  } |

**Java Solution**

|  |
| --- |
| import java.io.\*;  import java.util.\*;  public class Solution {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int t = sc.nextInt();  while(t-->0)  {  int n = sc.nextInt();  int q = sc.nextInt();  int arr[] = new int[n];  int prefix[] = new int[n];  for(int i=0;i<n;i++)  {  arr[i] = sc.nextInt();  }  // Calculating Prefix Sum Array  for(int i=1;i<n;i++)  {  prefix[i] = prefix[i-1] + arr[i];  }  int sum = 0;  // Running for Q Queries  for(int i=0;i<q;i++)  {  int l = sc.nextInt();  int r = sc.nextInt();  sum += prefix[r] - prefix[l-1];  }  if(sum % 2 == 0)  System.out.println("YES");  else  System.out.println("NO");  }  }  } |

**Python Solution**

|  |
| --- |
| t = int(input())  for \_ in range(t):  n, q = map(int, input().split())  arr = list(map(int, input().split()))  prefix = [0 for i in range(n)]  # Calculate Prefix Sum Array  for i in range(1, n):  prefix[i] = prefix[i-1] + arr[i]  sum = 0  for \_ in range(q):  l, r = map(int, input().split())  sum += prefix[r] - prefix[l-1]  if sum % 2 == 0:  print("YES")  else:  print("NO") |