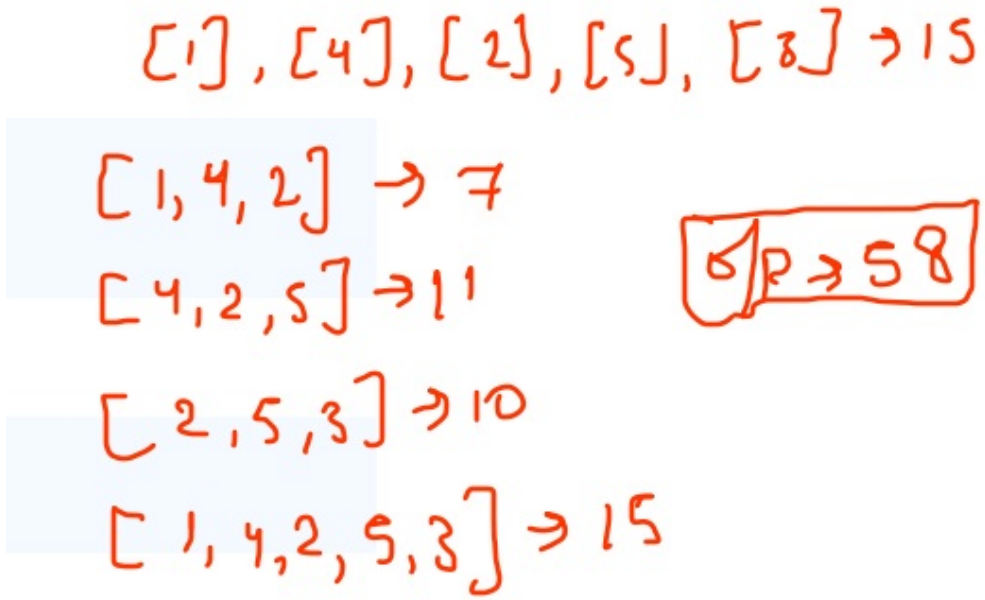
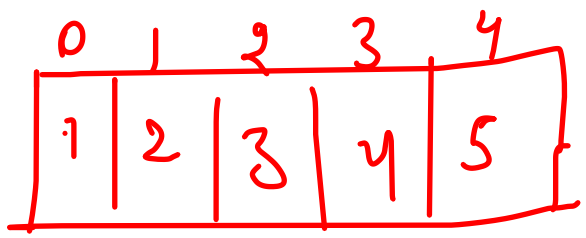
HW_Sum of All Odd Length Subarrays 1



arr →



n = 5

sum = 0 3 + 8 = 11

i = 0

arr[0] = 1

start = i+1 = 0+1 = 1

end = n-i = 5-0 = 5

total subarray = 1 x 5 = 5

odd subarray = (total+1)/2

= (5+1)/2 = 3

i = 1

arr[1] = 2

start = 1+1 = 2

end = 5-1 = 4

total = 2x4 = 8

odd = (8+1)/2 = 4

Sum = 2x4 = 8

total sum = odd subarray x arr[i]

3 x 1 = 3

[1,2], [1,2,3], [1,2,3,4], [1,2,3,4,5], [2,3,4], [2,3,4,5], [2,3], [2]

i = 2

arr[2] = 3

start = 2+1 = 3

end = 5-2 = 3

total = 3×3 = 9

odd = (9+1)/2 = 5

sum = 3×5 = 15

total sum = 11+15 = 26

```java
Scanner s = new Scanner(System.in);
int n = s.nextInt();

int arr[] = new int[n];
for(int i = 0; i <n;i++){
    arr[i] = s.nextInt();
}

int totalSum = 0;
for(int i = 0; i < n; i++){
    // each elem arr[i] will be contributing to several arrays
    // no. of subarrays before or at arr[i]
    int start = i + 1;

    // no. of subarrays after or at arr[i]
    int end = n - i;

    int totalSubarrays = start * end;
    // odd length sub arrays
    int oddsubArray = (totalSubarrays + 1)/2;

    totalSum += arr[i] * oddsubArray;
}

    System.out.println(totalSum);
}
```

TC → O(n)

SC → O(1)

O(log n) → O(n) → O(n log n) → O(n²) → O(n³) → O(2ⁿ)

Awesome — fine — ok-ok — bad — worst — horrible.