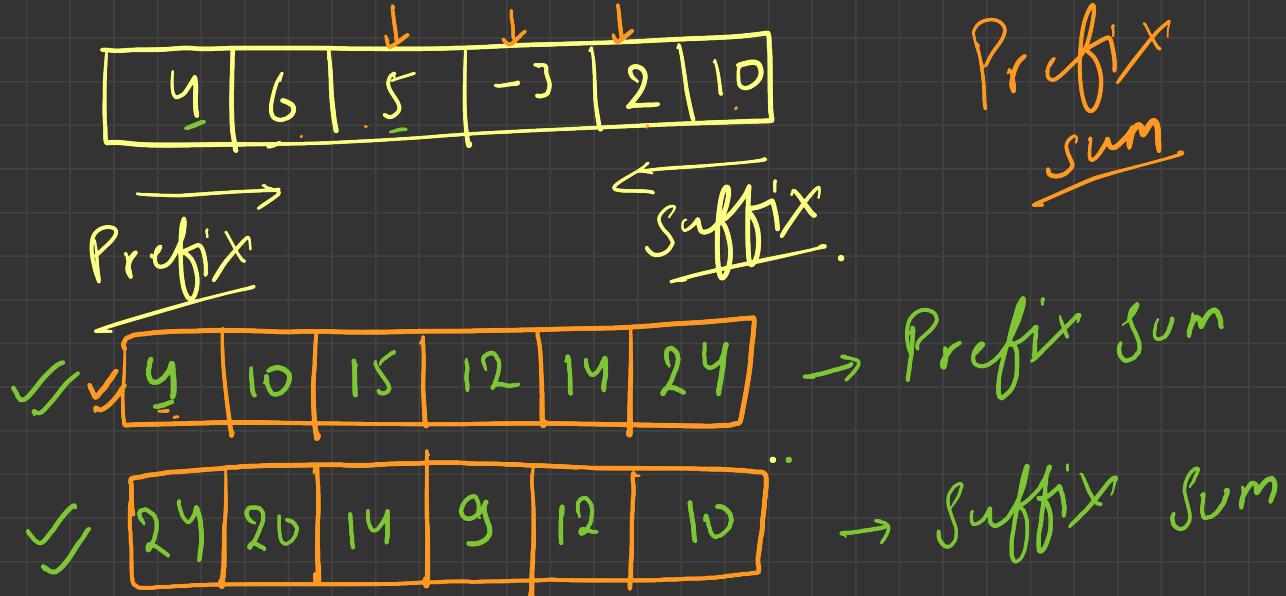


# Prefix and Suffix



Prefix Sum

// vector <int> prefix (n)

prefix[0] = arr[0];

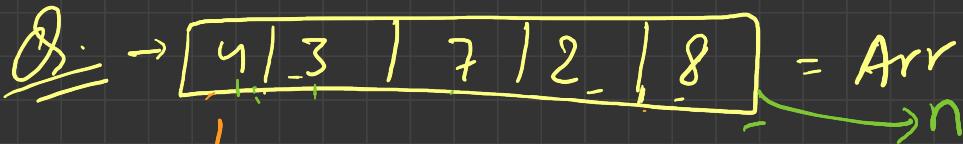
for (i = 1 ; i < n ; i++) {

// prefix[i] = prefix[i-1] + arr[i];

}

H.W Suffix Sum Code

loop



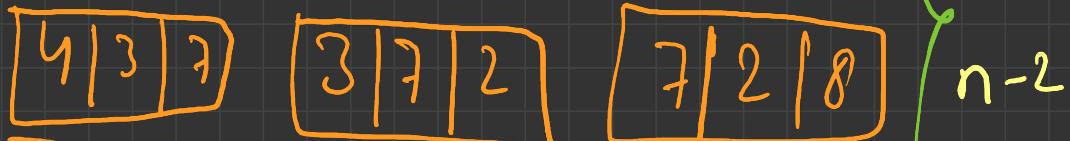
{ ✓ 1's size :



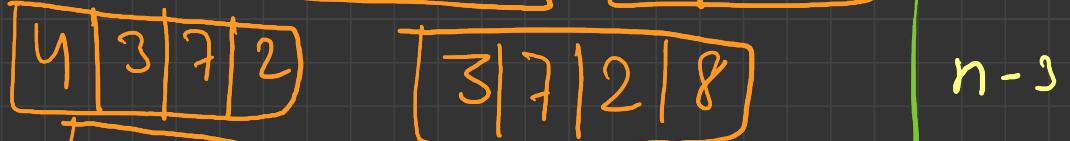
✓ 2's size :



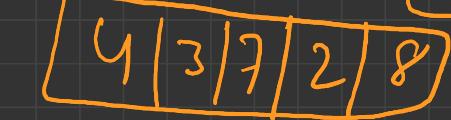
✓ 3's size :



✓ 4's size :



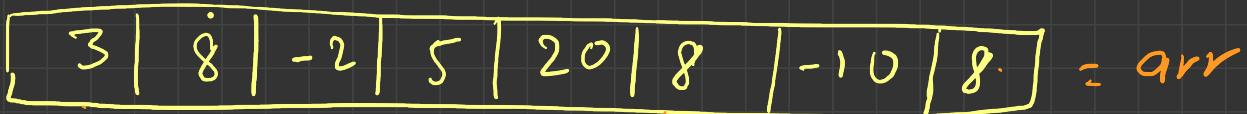
✓ 5's size :



n's size

→ 1

## Divide array in 2 Subarray with equal sum

 = arr

```
for (i=0 ; i<n-1 ; i++) { → n n  
    int sum1=0 , sum2=0 ;  
    for ( j=0 ; j<=i ; j++) { → n X } 2n  
        sum1 += arr[j] ;  
    for ( j=i+1 ; j<n ; j++) { → n }  
        sum2 += arr[j] ;  
    if (sum1 == sum2)  
        return yes
```

The diagram illustrates the time complexity of the algorithm. It shows nested loops and conditional statements with associated time complexities:  $n$  for the outer loop,  $n$  for the first inner loop,  $n$  for the second inner loop, and  $n^2$  for the if condition. A large bracket on the right side groups all these components together, labeled with  $O(n^2)$ .

3	8	-2	5	20	8	-10	8
							= arr

Total sum = 0;

```
for(i=0; i < n; i++) { }
```

TotalSum += arr[i];

```
int prefix = 0;
```

```
for (i=0; i < n-1; i++) { }
```

prefix += arr[i];

ans = totalSum - prefix;

if (ans == prefix) { }

return 1;

return 0;

n

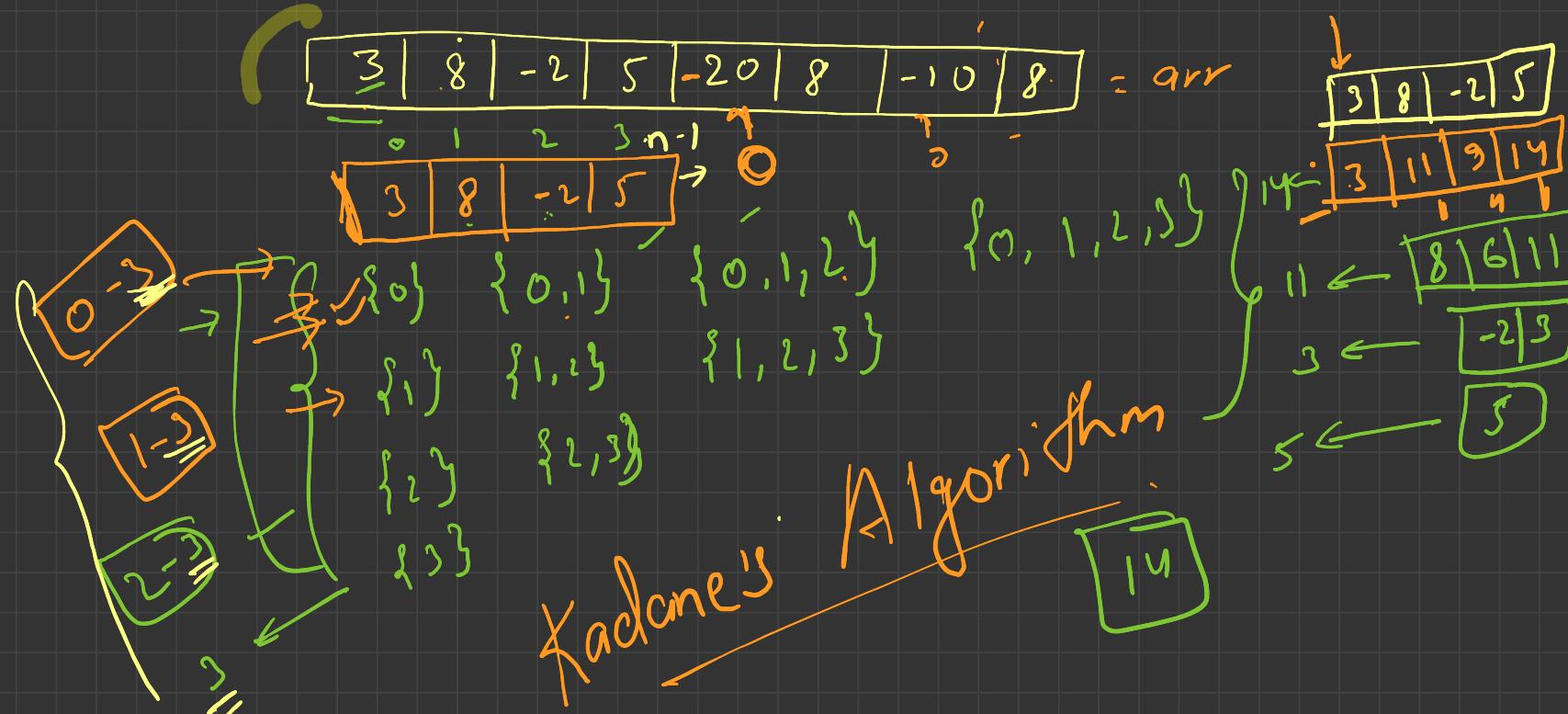
n + n

O(n)

Prefix = Total-prefix

{ } } } }

# Largest Sum Contiguous Subarray

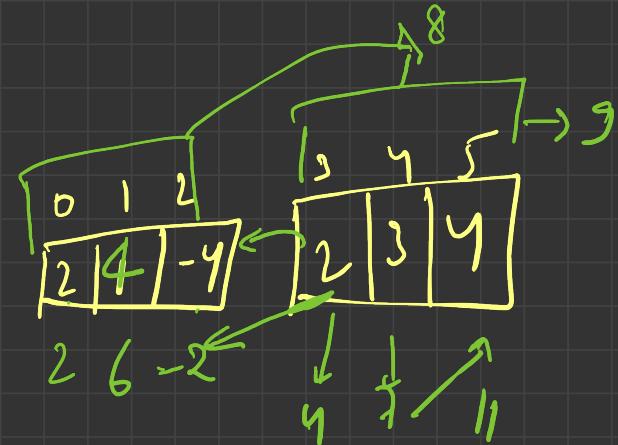


$\max = INTMIN$  { }  $\rightarrow \eta$   
 for (  $i=0$ ;  $i < n$ ;  $i++$  )  
 .  
 Prefix = 0  
 for (  $j=j$ ;  $j < n$ ;  $j++$  )  
 prefix += arr [j];  
 $\max = \max (\max, \text{prefix})$ ;  
 }  
 return max;

$n \times n$   
 $O(n^2)$

4	-6	3	7
4	-2	↓	↓
0	3	10	

maxi = INT - MIN  $\neq 10$



3	8	-2	5	20	8	-10	8
---	---	----	---	----	---	-----	---

 = arr

Maxi : INT-MIN    Prefix : 0;

for (i=0; i<n; i++) {  $\rightarrow O(N)$

    prefix += arr[i]

    maxi = max (maxi, prefix) .

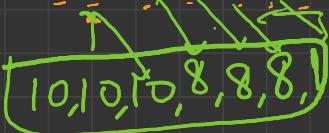
    if (prefix < 0) {

        prefix = 0 ;

$\text{ans} = \text{INT} - \text{MIN}$

## Maximum Difference between Two Elements

$\text{arr} = \{2, 3, 10, 6, 4, 8, 1\}$



```
for (i=0; i<(n-1); i++) {  
    for (j=i+1; j <(n); j++) {  
        ans = max (ans, arr[j] - arr[i])  
    }  
}
```

$O(N^2)$

## Maximum Difference between Two Elements

$arr = \{2, 3, 10, 6, 4, 8, 1\} \longrightarrow$



suffix

suffix  $\rightarrow 8$

$$8 - 1 = ?$$







