Subarrays Content

- Subarray Basics
- Printing a subarray
- Generating all subarrays
- Print all subarray sums.
  - Approach 1
  - Approach 2
- Max Subarray Sum
- Sum of all subarray sums → A1, A2

Subarray Basics Revision

- Continuous part of an array is called subarray.
- Single element is an Subarray. ✓
- Complete array is an subarray. ✓

Ex

	0	1	2	3	4	5	6	7	8	9
arr:	-2	4	6	3	8	1	4	3	2	-10

① indices : [3 4 5 7 8] : Not continuous.

② indices : [4 5 6 7 8] : → [4 8] → All indices from 4-8.

③ indices : [2 6] : 2 3 4 5 6

start index, end index

```
printSubArray(arr[], start, end]){
    // subarray[start, end]
    for(int i = start ; i <= end ; i++){
        print(arr[i]);
    }
}
```

TC : O(N)
SC : O(1)

How many subarrays?

Ex 1 :- arr : [-1 (0) | 3 (1) | 2 (2) | 3 (3)] ← N = 4

① [0 - 0] → {-1} → 1

② [0 - 1] → {-1, 3} → 2

③ [0 - 2] → {-1, 3, 2} → 4

④ [0 - 3] → {-1, 3, 2, 3} → 7

⑤ [1 - 1] → {3} → 3

⑥ [1 - 2] → {3, 2} → 5

⑦ [1 - 3] → {3, 2, 3} → 8

⑧ [2 - 2] → {2} → 2

⑨ [2 - 3] → {2, 3} → 5

⑩ [3, 3] → {3} → 3

1 + 4 + 7 + 5 + 8 + 2 + 5 + 3 ⇒ 38

Total No of subarrays = 4 + 3 + 2 + 1 = (10) subarrays

⇒ $\frac{4 * 5}{2} = 10$

// Given N elements, How many subarrays.?

arr :- {0 1 2 3 4 ..... N-2 N-1}

Start 0	Start 1	Start 2	...	Start N-2	Start N-1
0 - 0	[1 - 1]	2 - 2			
0 - 1	[1 - 2]	2 - 3			
0 - 2	[1 - 3]	2 - 4		2 subarrays	1
0 - 3					
0 - 4					
⋮	⋮	⋮			
0 - N-1	[1 .. N-1]	2 - (N-1)			
N subarrays	N-1 subarrays	(N-2) subarrays			

Total No of subarrays = N + (N-1) + (N-2) + - - - + 2 + 1

= $\frac{N * (N+1)}{2}$ ←

// Print all subarray.

start of subarray.

```
for(start = 0; start < N; start++){
    for(end = start; end < N; end++){
        // [start, end] is a subarray.
        {
            for(k = start; k <= end; k++){
                print(arr[k]);
            }
        }
        print("\n");
    }
}
```

TC : $O(N^3)$, SC : $O(1)$

N = 4

0	1	2	3
8	2	9	10

arr: [8 2 9 10]

[0-0] → {8}

[0-1] → {8, 2}

[0-2] → {8, 2, 9}

[0-3] → {8, 2, 9, 10}

[1-1] → {2}

[1-2] → {2, 9}

[1-3] → {2, 9, 10}

[2-2] → {9}

[2-3] → {9, 10}

[3-3] → {10}

// Print all subarray sums.

TC : O(N^3), SC : O(1)

```
for(start = 0; start < N; start++){        // start of subarray.
    for(end = start; end < N; end++){
        sum = 0                            // [start, end]
        for(k = start; k <= end; k++){
            sum = sum + arr[k];
        }
        print(sum);
    }
}
```

Optimize this?

① Using Prefix sum. → TC : O(N^2), SC : O(N)

N + N^2

Build PS | Print sum of each subarray.

PF[end] - PF[s-1]

// Print all subarrays sums starting at index 2.

	0	1	2	3	4	5	6	7
Ex: arr :	7	3	2	-1	6	8	2	3

sum = 0

Start 2

[2, 2] : 2

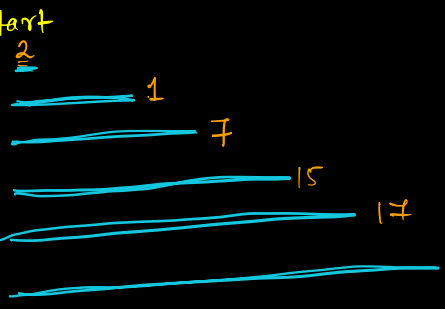
[2, 3] : 1

[2, 4] : 7

[2, 5] : 15

[2, 6] : 17

[2, 7] : 20



s = 0

end = 2 s : 2

end = 3 s = 1

end = 4 s = 7

end = 5 s = 15

end = 6 s = 17

end = 7 s = 20

```
sum = 0
for(end = 2; end < N; end++){
    sum = sum + arr[end];
    print(sum);
}
```

// Print all subarray sums using Carry forward.

TC : O(N²)
SC : O(1)

```
for (start = 0; start < N; start++) {
    sum = 0
    for (end = start; end < N; end++) {
        sum = sum + arr[end];
        print(sum);
    }
}
```

All subarrays starting from index i

↳ each sum is a subarray sum.

// Print max subarray sum

TC : O(N²)
SC : O(1)

```
maxsum = arr[0]
for (i = 0; i < N; i++) {
    sum = 0
    for (j = i; j < N; j++) {
        sum = sum + arr[j]; // subarray sum
        if (sum > maxsum) {
            maxsum = sum;
        }
    }
}
return maxsum;
```

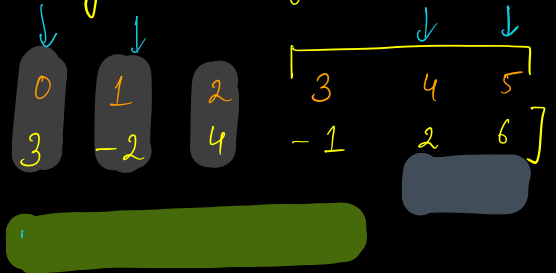
→ Kadane's Algorithms
TC O(N) & O(1) SC

// Print sum of all subarray sums

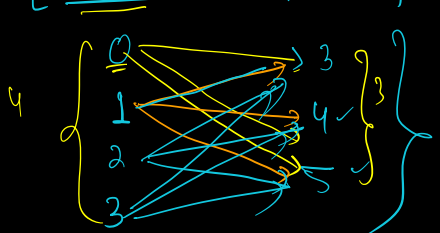
```
totalsum = 0
for(i = 0; i < N; i++){
    sum = 0
    for(j = i; j < N; j++){
        sum = sum + arr[j]; //sum is subarray sum
        totalsum = totalsum + sum;
    }
}
return totalsum;
```

TC : O(N^2)
SC : O(1)

Q1. In how many subarrays index 3 is present ?



{ Start end }



4 * 3 = 12 subarrays

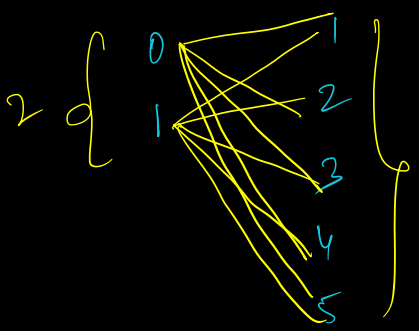
[0 ... 3,4,5] = 3
[1 ... 3,4,5] = 3
[2 ... 3,4,5] = 3
[3 ... 3,4,5] = 3

12 subarr

Quiz1 In how many subarrays index 1 is present ?

arr :- [3, -2, 4, -1, 2, 6]

Start end



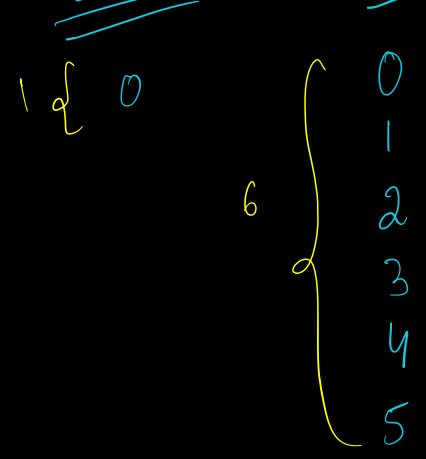
$\Rightarrow 2 * 5$

$\Rightarrow$ 10 subarrays. ??

Quiz2 : In how many subarrays index 0 is present ?

arr : [3, -2, 4, -1, 2, 6]

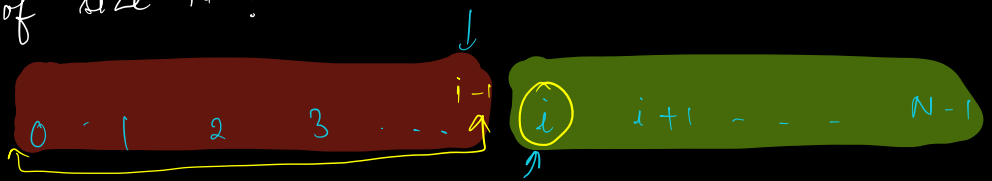
Start end



$\Rightarrow 1 * 6$

$\Rightarrow$ 6 subarrays

Q In how many subarrays index i is present in an array of size N ?



Start $\Rightarrow [0 \ldots i] \Rightarrow i - 0 + 1 \Rightarrow (i+1)$

end $\rightarrow [i \ldots N-1] \Rightarrow N-1-i+1 \Rightarrow (N-i)$

$= \boxed{(i+1) * (N-i)}$

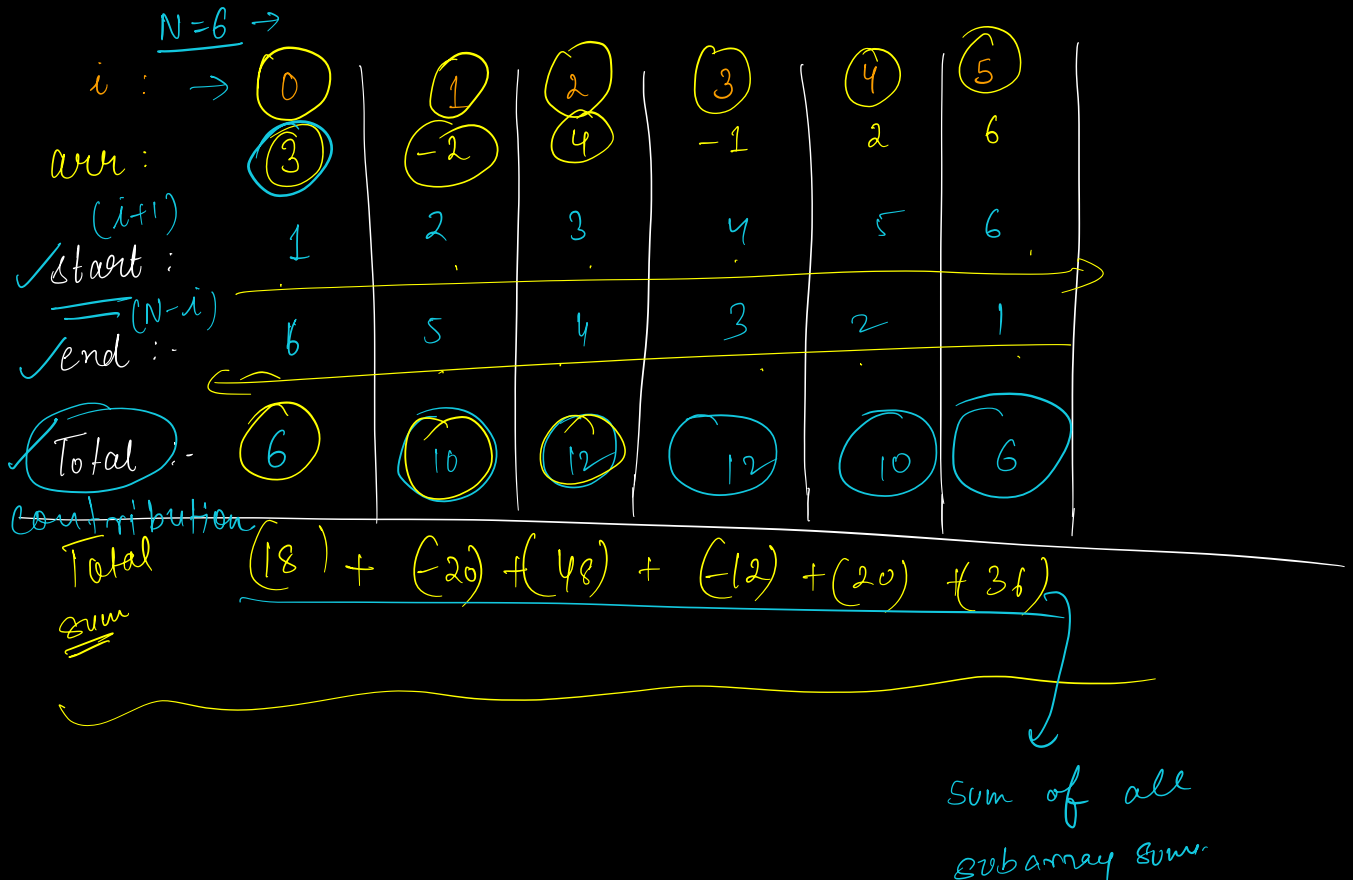
$N = 4$

$\begin{cases} \text{Start} = i+1 \\ \text{end} = (N-i) \\ \quad\quad 4-3 \end{cases}$

i :-	0	1	2	3
arr :-	-1	3	2	3
start :-	1	2	3	4
end :-	4	3	2	1
Total :-	4	6	6	4
	-4	18	12	12

Total $\Rightarrow -4 + 18 + 12 + 12 \Rightarrow 38$

Total sum of all Subarray sums


N=6 →
i : → 0 1 2 3 4 5
arr : 3 -2 4 -1 2 6
(i+1)
start : 1 2 3 4 5 6
(N-i)
end : 6 5 4 3 2 1
Total :- 6 10 12 12 10 6
Contribution
Total sum
(18) + (-20) + (48) + (-12) + (20) + (36)
Sum of all subarray sum.

// Sum of all subarray sums

```
totalsum = 0
for(i=0; i<N; i++){
    start = i+1   // possible no of indices at which subarray can start
    end = N-i     // possible indices at which subarray can end.
    total_subarrays = start * end; // total No of subarrays in which index i is present

    contribution = arr[i] * total_subarrays
    total_sum = total_sum + contribution.
}
```

TC : O(N)

SC : O(1)