

Today's Content

1. Subarrays Introduction
2. Print all subarrays from $s..e$
3. Print all subarray sums
 - 3a Optimization using Prefix Sum
 - 3b Optimization using Contribution Technique

Introduction to Subarrays:

Definition:

Subarray is continuous part of an array

1 element or complete array is also considered as subarray.

Note: Subarray considered left to right

0 1 2 3 4 5 6 7 8
arr[] = { 4 1 2 3 -1 6 9 8 12 }

{ 2 3 -1 6 } : Yes

{ 9 } : Yes

{ 4 1 2 3 -1 6 9 8 12 } : Yes

{ 4 2 } No

{ 1 2 6 } No

Representation of Subarray:

With start & end index of subarray

0 1 2 3 4 5 6 7 8
arr[] = { 4 1 2 3 -1 6 9 8 12 }

Start = end =

3 8 { 3 -1 6 9 8 12 }

2 5 { 2 3 -1 6 }

How many subarrays start from index: 0

Ex: $arr[] = \{ \overset{0}{4} \overset{1}{2} \overset{2}{3} \overset{3}{7} \overset{4}{8} \}$

#start:

$[0, 0] = \{4\}$

#start: 1

$[0, 1] = \{4, 2\}$

$[1, 1] = \{2\}$

$[0, 2] = \{4, 2, 3\}$

$[1, 2] = \{2, 3\}$

$[0, 3] = \{4, 2, 3, 7\}$

$[1, 3] = \{2, 3, 7\}$

$[0, 4] = \{4, 2, 3, 7, 8\}$

$[1, 4] = \{2, 3, 7, 8\}$

#Total subarrays:

$arr[] = \{ \overset{0}{4} \overset{1}{2} \overset{2}{3} \overset{3}{7} \overset{4}{8} \}$
→
→
→
→ } #Sub = 5

#Total subarrays len = 5

$$5 + 4 + 3 + 2 + 1 = 15$$

$arr[] = \{ \overset{0}{4} \overset{1}{2} \overset{2}{3} \overset{3}{7} \overset{4}{8} \}$
→
→
→ } #Sub = 4

#Total subarrays len = N

$arr[] = \{ \overset{0}{4} \overset{1}{2} \overset{2}{3} \overset{3}{7} \overset{4}{8} \}$
→
→ } #Sub = 3

$$N + N-1 + N-2 + \dots + 1 = \frac{(N)(N+1)}{2}$$

$arr[] = \{ \overset{0}{4} \overset{1}{2} \overset{2}{3} \overset{3}{7} \overset{4}{8} \}$
→ } #Sub = 2

$arr[] = \{ \overset{0}{4} \overset{1}{2} \overset{2}{3} \overset{3}{7} \overset{4}{8} \}$
→ } #Sub = 1

Q0: Print a given subarray

#start #end

```
void printSub(vector<int> &arr, int i, int j) { Tc: O(N) sc: O(1)
    for (int k = i; k <= j; k++) {
        print(arr[k] + " ");
    }
}
```

Q1: Print all subarrays.

Ex: arr[4] = { 3 7 2 9 }

arr[N] = { 0 1 2 ... [i] [i+1] [i+2] ... N-1 }

#Subarrays Output

{ 0 0 }	: { 3 }
{ 0 1 }	: { 3 7 }
{ 0 2 }	: { 3 7 2 }
{ 0 3 }	: { 3 7 2 9 }
{ 1 1 }	: { 7 }
{ 1 2 }	: { 7 2 }
{ 1 3 }	: { 7 2 9 }
{ 2 2 }	: { 2 }
{ 2 3 }	: { 2 9 }
{ 3 3 }	: { 9 }

```
void printAll(vector<int> &arr) {
    int N = arr.size();
    for (int i = 0; i < N; i++) { i: start
        for (int j = i; j < N; j++) { j: end
            // subarray [i..j]
            for (int k = i; k <= j; k++) {
                print(arr[k]);
            }
            print("\n");
        }
    }
}
```

Tc: $\# O(N^2) * N = O(N^3)$

Calculate & Return sum of all subarray sums.

Ex: $arr[3] = \begin{matrix} 0 & 1 & 2 \\ 3 & 4 & 2 \end{matrix}$

Subarrays:		#Sum
[0 0]	{3} →	3
[0 1]	{3 4} →	7
[0 2]	{3 4 2} →	9
[1 1]	{4} →	4
[1 2]	{4 2} →	6
[2 2]	{2} →	2
Total =		31

#ideal: generate all subarray sums & add in total & return it.

```
long printAll(vector<int> &arr) {
    long tsum = 0;
    int N = arr.size();
    for (int i = 0; i < N; i++) { i: start
        for (int j = i; j < N; j++) { j: end
            // subarray [i..j]
            long sum = 0; // 1 subarray sum
            for (int k = i; k <= j; k++) {
                sum = sum + arr[k]
            }
            tsum = tsum + sum;
        }
    }
    return tsum;
}
```

TC: $\# O(N^2) * O(N) = O(N^3)$ SC: $O(1)$

↓
 N^2 subarrays

↓
iterate & get sum for 1 subarray

#idea2 generate all subarray sums using pfsun & add in total.

long printAll(vector<int> &ar) { Tc: $O(N + N^2)$ = $O(N^2)$ Sc: $O(N)$

int N = ar.size();

generate pf

pf[]

generate all subarray sums

long pf[N], sum = 0;

for(int i = 0; i < N; i++) {

sum = sum + ar[i];

pf[i] = sum;

}

long tsum = 0;

for(int i = 0; i < N; i++) { i: start

for(int j = i; j < N; j++) { j: end

// subarray [i..j] using pf[] = pf[j] - pf[i-1];

if(i == 0) { // [0..j]

tsum += pf[j];

else {

tsum += pf[j] - pf[i-1];

}

}

return tsum;

}

Ideas: In Question, Calculate sum of all we apply Contribution Technique
Contribution Technique = Add contribution of each ele in final ans.

Ex: $arr[3] = \begin{matrix} 0 & 1 & 2 \\ \{3 & 4 & 2\} \end{matrix}$

Subarrays:

$\begin{bmatrix} [0, 0] & \{3\} \uparrow \\ [0, 1] & \{3, 4\} \uparrow \\ [0, 2] & \{3, 4, 2\} \uparrow \\ [1, 1] & \{4\} \uparrow \\ [1, 2] & \{4, 2\} \uparrow \\ [2, 2] & \{2\} \uparrow \end{bmatrix}$

Contribution:

ele occurrence contribution

$$3 * 3 = 9$$

$$4 * 4 = 16$$

$$2 * 3 = 6$$

$$\text{Total Sum} = 31$$

Ex: $arr[4] = \begin{matrix} 0 & 1 & 2 & 3 \\ \{2 & 8 & -1 & 4\} \end{matrix}$

Subarrays:

$\begin{bmatrix} [0, 0] & \{2\} \\ [0, 1] & \{2, 8\} \\ [0, 2] & \{2, 8, -1\} \\ [0, 3] & \{2, 8, -1, 4\} \\ [1, 1] & \{8\} \\ [1, 2] & \{8, -1\} \\ [1, 3] & \{8, -1, 4\} \\ [2, 2] & \{-1\} \\ [2, 3] & \{-1, 4\} \\ [3, 3] & \{4\} \end{bmatrix}$

Contribution:

ele occurrence contribution

$$2 * 4 = 8$$

$$8 * 6 = 48$$

$$-1 * 6 = -6$$

$$4 * 4 = 16$$

$$\text{Total Sum} = 66$$

Catch:

To calculate contribution of element we need to get its occurrences.

Occurrences: In this problem, it is no. of times $arr[i]$ comes in subarray.

In how many subarrays a particular index i will be present

Ex: $ar[6] = \{ 3 \quad -2 \quad 4 \quad -1 \quad 2 \quad 6 \}$

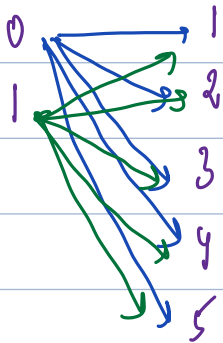
Subarrays: $\{s..e\}$

#start = $\checkmark \quad \checkmark \quad \times \quad \times \quad \times \quad \times$

#end = $\times \quad \checkmark \quad \checkmark \quad \checkmark \quad \checkmark \quad \checkmark$

In how many subarrays index i is present

startind endind Total Subarrays



$$2 * 5 = 10.$$

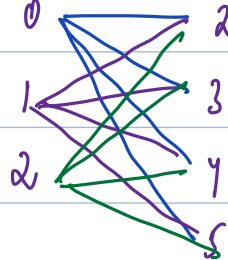
In how many subarrays index 2 is present

Ex: $ar[6] = \{ 3 \quad -2 \quad 4 \quad -1 \quad 2 \quad 6 \}$

startind endind Total Subarrays

#start = $\checkmark \quad \checkmark \quad \checkmark \quad \times \quad \times \quad \times$

#end = $\times \quad \times \quad \checkmark \quad \checkmark \quad \checkmark \quad \checkmark$



$$3 * 4 = 12$$

In how many subarrays index i is present

$ar[N] = \{ a_0 \quad a_1 \quad a_2 \quad \dots \quad a_{i-1} \quad a_i \quad a_{i+1} \quad a_{i+2} \quad \dots \quad a_{N-2} \quad a_{N-1} \}$ $a \quad b \quad b-a+1$

#start = $\{ \checkmark \quad \checkmark \quad \checkmark \quad \dots \quad \checkmark \quad \checkmark \quad \times \quad \times \quad \dots \quad \times \quad \times \}$ # $[0..i] = i+1$

#end = $\{ \times \quad \times \quad \times \quad \dots \quad \times \quad \checkmark \quad \checkmark \quad \checkmark \quad \dots \quad \checkmark \quad \checkmark \}$ # $[i..N-1] = N-i$

#subarrays = $\text{start} * \text{end} = (i+1)(N-i)$

#Tracing:

	i	0	1	2	3
N=4 ar[4] =		2	8	-1	4
#(i+1) =		1	2	3	4
#(N-i) =		4	3	2	1
#occurency =		4	6	6	4
#Contribut =		8 + 48	- 6 + 16	= 66	

```
long subSum(vector<int> &ar) { T: O(N) S: O(1)
```

```
    int N = ar.size();
```

```
    long sum = 0;
```

```
    for (int i = 0; i < N; i++) {
```

```
        long occ = (i+1)(N-i)
```

```
    } sum = sum + ar[i]*occ;
```

```
    return sum;
```