

Today's content

- Check if there exists a pair (i, j) such that $A[i] + A[j] == k$ && $i \neq j$
- Get no. of distinct elements in every window of size = k
- Find length of longest subarray with $\text{sum} = 0$

1. Given N array elements, check if there exists a pair (i, j) such that $arr[i] + arr[j] = k$ and $i \neq j$ → bool

$arr[]$:

0	1	2	3	4	5	6	7	8	9
8	9	1	-2	4	5	11	-6	7	5

	i	j	$arr[i]$	$arr[j]$	
$k = 11$	4	8	4	7	T
$k = 6$	2	5	1	5	T
$k = 22$					F

$$8 + \underline{\quad} = 11$$

↓

$$11 - 8$$

Idea 1: Check all unique pairs

$$a + b = k$$

$$b = k - a$$

```

for (i = 0; i < n; i++) {
    a = arr[i], b = k - arr[i]
    for (j = i + 1; j < n; j++) {
        if (arr[i] + arr[j] == k)
            return true
    }
}

```

TC: $O(N^2)$

return false

$arr[]$:

0	1	2	3	4	5	6	7	8	9
8	9	1	-2	4	5	11	-6	7	5

Idea 2: Create Hashset $\langle \text{int} \rangle$ hs
Insert entire data in hs

$\langle 8, 9, 1, -2, 4, 5, 11, -6, 7, 5 \rangle$

$k=11$

a	b	is b in HS
8	3	X
9	2	X
1	10	X
-2	13	X
4	7	✓

return True

$k=22$

a	b	is b in HS
8	14	X
11	11	✓

True X

$a == b$

arr[]: 0 1 2 3 4 5 6 7 8 9

8 9 1 -2 4 5 11 -6 7 5

Obs: If we know freq, we can solve prob

1. Hashmap $\langle \text{int}, \text{int} \rangle$ hm

arr[i] → freq

2. Iterate & insert in hm

{ 8:1, 9:1, 1:1, -2:1, 4:1, 5:2, 11:1, -6:1, 7:1 }

$k=22$

a	b	is b in HM
8	14	X
9	13	X
1	21	X
-2	24	X

4	18	X
5	17	X
11	11	X

$\text{if}(a == b) \text{freq}(b) > 1$

Ex 7 5 2 5 9 5 $k=10$

HM { 7:1 5:3 2:1 9:1 }

a	b	$\rightarrow k-a$	is b in HM
7	3		X
5	5		$\text{if}(\text{freq}(5) > 1) \checkmark$

Pseudocode

1. Create hm & insert arr[] $\rightarrow n$
2. for $i=0; i < N; i++$ $\leftarrow n$

a = arr[i]

b = k - a

if (a != b) <

if (b is in HM)

return true

else < // a == b

if (freq(b) > 1)

return true

return false

1 iter $\rightarrow O(1)$

$n \rightarrow n \times O(1)$

TC: $O(n)$

SC: $O(n)$

arr: 0 1 2 3 4 5 6 7 8 9 10 11
 8 9 1 -2 4 5 11 -6 7 5 11 11

↑
a

Idea: when we are at i^{th} index, all elements from $[0, i-1]$ in HS

$k=22$

a	b $\rightarrow k-a$	HS	
8	14	< 7 >	✗ Insert 8
9	13	< 8 7 >	✗ Insert 9
1	21	< 8, 9 >	✗ Insert 1
-2	24	< 8, 9, 1 >	✗ Insert -2
4	18	< 8, 9, 1, -2 >	✗ Insert 4
5	17	< 8, 9, 1, -2, 4 >	✗ Insert 5
11	11	< 8, 9, 1, -2, 4, 5 >	✗

$k=12$

a	b	HS	
8	4	< 7 >	✗
9	3	< 8 7 >	✗
1	11	< 8, 9 >	✗
-2	14	< 8, 9, 1 >	✗
4	8	< 8, 9, 1, -2 >	✓ T

Pseudocode -

```
hashset <int> hs;
```

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

```
    a = ar[i]
```

```
    b = k - a
```

```
    if (b in HS)
```

```
        return true
```

```
    hs.insert(a)
```

```
}
```

```
return false
```

TC: $O(N)$

SC: $O(N)$

Q. calculate no. of i, j such that
 $ar[i] + ar[j] = k$

TODO

hashmap

elem, freq



Q. Given N array elements, check if there exists a pair (i, j) such that $ar[i] - ar[j] = k$ and $i \neq j$

2. Given N array elements, calculate no. of distinct elements in every subarray of size K .

arr: 2 4 3 8 3 9 4 9

$K = 4$

	Distinct
[0-3]	4
[1-4]	3
[2-5]	3
[3-6]	4
[4-7]	3

Idea:
for every window, get no. of distinct elements

(K)
[0 - N]
[0 K-1]
[1 K]
[2 K+1]
[3 K+2]
⋮
[N-K N-1]

print no. of distinct element in every window of size = K

HashSet <int> hs;

for (s=0; s <= N-K; s++)

<
for (j=s; j < s+K; j++)
| hs.insert(arr[j])
>
print hs.size()
hs.clear()
>

Iterations

$(N-K+1) * K$ } Max

$K \rightarrow [1 \quad N]$

if $K = N/2$

$(N - N/2 + 1) * N/2$

$= (N/2) (N/2)$

$\rightarrow TC: O(N^2)$

$SC: O(K)$

$\rightarrow O(N)$

arr[] : 0 1 2 3 4 5 6 7
 2 4 3 8 3 9 4 9

k=4

	Remove	Add	HS	no.
Initially [0-3]			<2, 4, 3, 8>	4
[1-4]	0 th	4 th	<4, 3, 8>	3
[2-5]	1 st	5 th	<3, 8, 9>	3
[3-6]	2 nd	6 th	<8, 9, 4>	3

Obs : If we remove an ele, indirectly all occurrences of same element is also removed.

arr[] : 0 1 2 3 4 5 6 7
 2 4 3 8 3 9 4 9

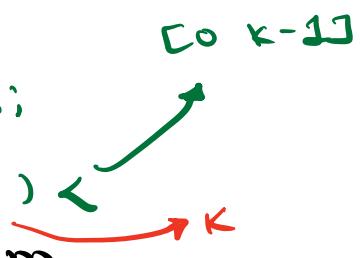
	Remove	Add	HIM
[0-3]			<2:1, 4:1, 3:1, 8:1> ⇒ 4
[1-4]	0 th	4 th	del 2 <2:0, 4:1, 3:2, 8:1> ⇒ 3
[2-5]	1 st	5 th	del 4 <4:0, 3:2, 8:1, 9:1> ⇒ 3
[3-6]	2 nd	6 th	<3:1, 8:1, 9:1, 4:1> ⇒ 4
[4-7]	3 rd	7 th	del 8 <8:0, 3:1, 9:2, 4:1> ⇒ 3

In hm, if freq = 0, remove from hm

Pseudocode

Given N elements, K

HashMap <int, int> hm;


for (i = 0; i < K; i++) 
 // ar[i] insert in hm
 if (ar[i] is in hm) <
 hm[ar[i]] ++
 >
 else <
 hm.insert(ar[i], 1)
 >
>

print (hm.size())

$$\text{len} = e - s + 1$$

$$K = e - s + 1$$

$$K + s - 1 = e$$

[s e]
 
 K

```

for (s=1 ; s <= N-k ; s++) <
    // subarray [s e]  $\rightarrow N-k$ 

    e = s + k - 1

    // rm s-1, add e  $k + N - k = N$ 

    hm[ar[s-1]]--;
    if (hm[ar[s-1]] == 0) <
        | hm.remove(ar[s-1])
        >

    if (ar[e] is in hm) <
        | hm[ar[e]]++;
        >
    else <
        | hm.insert(ar[e], 1);
        >

    print (hm.size())
>

```

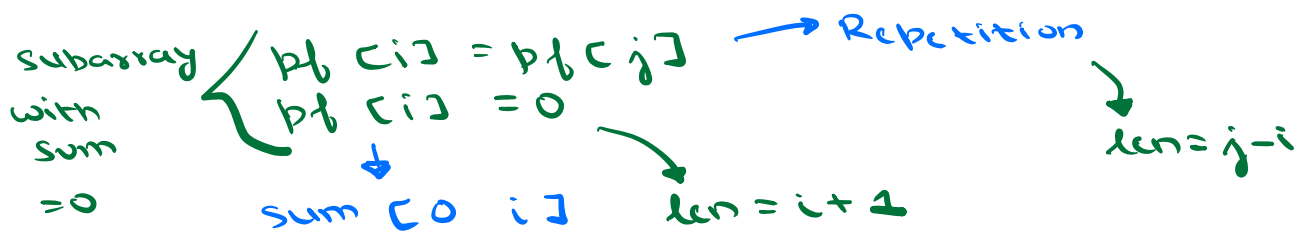
$T_C: O(N)$
 $SC: O(k)$
 N
 $(k=N)$

11:17

3. Given N array elements, find length of longest subarray sum=0

arr : 3 3 4 -5 -2 2 1 -3 3 -1 5 -4

pf : 3 6 10 5 3 5 6 3 6 5 10 6



In case of repetition, consider farthest index → (first, last index)

	0	1	2	3	4	5	6	7	8	9	10	11
arr	3	3	4	-5	-2	2	1	-3	3	-1	5	-4
pf	3	6	10	5	3	5	6	3	6	5	10	6
curLen					4	2	5	7	7	6	8	10
maxLen					4	4	5	7	7	7	8	10

HM \rightarrow pf sum, 1st index

<3, 0>
 <6, 1>
 <10, 2>
 <5, 3>

pf [i] in HM \rightarrow repetition
 $len = i - hm[pf[i]]$

Edge case

	0	1	2	3
Ex	-2	2	1	-1
pf	-2	0	1	0
				2

ans = 4

Pseudocode

1. Create pf $\rightarrow N$
2. Hashmap <int, int> hm
 ans = 0

```

for (i=0; i < N; i++) {
    if (pf[i] == 0) {
        ans = max(ans, i+1)
        continue
    }
    if (pf[i] is in hm) {
        // repetition
        ans = max(ans, i - hm[pf[i]])
    }
    else {
        hm.insert(pf[i], i)
    }
}
return ans

```

HM <ele, first index>

<-2, 0>
 <0, 1>
 <1, 2>

TC: $O(N)$

SC: $O(N)$

pf \downarrow hm

Worst case $\rightarrow n$