Today's Content

→ Pair Sum = k
→ Distinct elements in every window of len = k.

Problem Solving → Sunday Morning → 10 am.

Today's Quote :-

'All of man's troubles arise because he can't sit in a room quietly by himself.' If you could just sit of 30 mins & be happy you are very successfull.

Ques) Given N array elements, check if there exists a pair (i, j) such that arr[i] + arr[j] = k & i != j

	0	1	2	3	4	5	6	7	8	9
arr[] =	8	9	1	-2	4	5	11	-6	7	5

k = 11 yes (i, j) = (4, 6)

k = 6 yes (i, j) = (2, 5)

k = 22 No.

a + b == k

Idea :- check for every pair sum = k.

T.C → O(N^2) S.C → O(1).

```
for (i=0; i<N; i++) {
    a = arr[i], b = k - a
    for (j=i+1; j<N; j++) {
        if (arr[j] == b) {
            return True
        }
    }
}
return false
```

	0	1	2	3	4	5	6	7	8	9
arr[] =	8	9	1	-2	4	5	11	-6	7	5

This will fail.

idea 2 :- Optimization using HashSet.

↳ Insert all the elements in hashset.

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

k = 11

a + b = 11

a	b(k-a)	
8	3	✗
9	2	✗
1	10	✗
-2	13	✗
4	7	✓

{ return true }

k = 5

a + b = 5

a	b(k-a)	
8	-3	✗
9	-4	✗
1	4	✓

k = 22

a + b = 22

a	b(k-a)	
8	14	✗
9	13	✗
1	21	✗
-2	24	✗

4	18	*
5	17	*
11	11	{return True}

idea 3 Optimization using HashMap<int, int>

↓ array ↓ freq

	0	1	2	3	4	5	6	7	8	9
arr[] =	8	9	4	-2	4	5	11	-6	7	5

k = 10

insert everything int hMap

HMap → <8,1>, <9,1> <4,2>, <-2,1>, <5,2>
<11,1>, <-6,1>, <7,1>

a + b = 22

a	b (k-a)	
8	14	*
9	13	*
4	18	*
-2	24	*
4	18	*
5	12	*
11	11	a == b

hm[a] > 1 since hm[11] = 1.

-6	28	*
7	15	*
5	12	*

(return false)

a + b = 10

a	b (k - a)
8	2
9	1
4	6
-2	12
4	6
5	5

a == b , freq(a) ≥ 1 , freq(5) ≥ 1

↓

we can get reqd sum.

T.C → O(N), S.C → O(N)

```
bool pairSum(int arr[], int k) {
    HMap <int, int> hm;          → T.C → O(N)
    Insert all arr[] → hm // Todo.
    for (i=0; i<n; i++) {        → T.C → O(N)
        a = arr[i], b = k - a
        if (a != b && hm.search(b) == T)
            return True
        } else if (a == b && hm[a] > 1){
            return True
        }
    }
    return false
}
```

idea 4:- Optimization using hashSet :-

	0	1	2	3	4	5	6	7	8	9
arr[] =	8	9	5	-2	11	5	4	-6	7	4

↳ Idea :- If we are at ith idx, insert only [0, i-1] elements in hset.

a + b = 10

a	b	Hs
8	2	{8}
9	1	{8, 9}
5	5	{8, 9, 5}
-2	12	{8, 9, 5, -2}
11	-1	{8, 9, 5, -2, 11}
5	5	{8, 9, 5, -2, 11}

↳ return true.

$a + b = 11$

a	b	Hy
8	3	{3}
9	2	{8}
4	7	{8, 9}
-2	13	{8, 9, 4}
4	7	{8, 9, 4, -2}
5	6	{8, 9, 4, -2}
11	0	{8, 9, 4, -2, 5}
-6	17	{8, 9, 4, -2, 5, 11}
7	4	{8, 9, 4, -2, 5, 11, -6}

$a + b = 22$

a	b	Hy
8	14	{3}
9	13	{8}
4	18	{8, 9}
-2	24	{8, 9, 4}
11	11	{8, 9, 4, -2}

↳ Not failing here.

Pseudo code :- T.C → O(N), S.C → O(N).

```
bool TargetSum( int arr[], int k) {
    int n = arr.length;
    HashSet<int> hs;
    for (i=0; i<n; i++) {
        a = arr[i], b = k - a ;
        if (hs.search(b) == True) {
            return True
        }
        hs.insert(a)
    }
    return false
}
```

Break 9:53 pm to 10:05 pm

Q0) Given N elements, calculate no. of distinct elements in every subarray of size = k.

		0	1	2	3	4	5	6	7	8	9
Ex:-	arr[10] =	2	4	3	8	3	9	4	9	4	10

k = 4

Subarray	Print
[0 - 3]	4
[1 - 4]	3
[2 - 5]	3
[3 - 6]	4
[4 - 7]	3
[5 - 8]	2
[6 - 9]	3

idea :-

for every subarray of len = k, insert all elements into hashset, and get no. of distinct elements.

$\Rightarrow$ $(n-k+1)\,k$ when $k = \frac{n}{2}$.

$\Rightarrow \left(n-\frac{n}{2}+1\right)\left(\frac{n}{2}\right) \Rightarrow \left(\frac{n^2}{T.C \rightarrow}\right)$

S.C $\rightarrow O(k)$.

first subarray

0

Last subarray.

$-x \Rightarrow n-k$

$[x \quad n-1] = k$

$n-x-1+1 = k$

$n-x = k$

$n-k = x$

$\Downarrow$

Total Subarrays $= n-k+1$

idea2 :- Optimization using hashset.

		0	1	2	3	4	5	6	7	8	9
Ex:-	arr[10] =	2	4	3	8	3	9	4	9	4	10

k = 4

Subarray

(0 - 3)	⟶	HS = {2, 4, 3, 8}	: 4
(1 - 4)	del arr[0] insert arr[4]	HS = {4, 3, 8}	: 3
(2 - 5)	del arr[1] ins arr[5]	HS = {3, 8, 9}	: 3
(3 - 6)	del arr[2] ins arr[6]	HS = {8, 9, 4}	: 3

Note :- In hset, deleting an element will indirectly delete all occurrences.

idea3 :- Optimization using hashMap.

		0	1	2	3	4	5	6	7	8	9
Ex:-	arr[10] =	2	4	3	8	3	9	4	9	4	10

k = 4

Subarray	HMap	
(0 - 3)	{<2,1> <4,1> <3,1> <8,1>}	: 4
(1 - 4)	del (0) add (4) {<4,1> <3,2> <8,1>}	: 3
(2 - 5)	del (1) add (5)	<3,2> <8,1> <9,1> : 3
(3 - 6)	del (2) add (6)	<3,1> <8,1> <9,1> <4,1> : 4

(4-7) del(3) add(7) <3,1><9,2><4,1> : 3

(5-8) del(4) add(8) <9,2><4,2> : 2

(6-9) del(5) add(9) <9,1><4,2><10,1> : 3

(s e) → del(s-1), add(e)

→ T.C → O(N) S.C → O(K).

```
void distinctMap ( int arr[], int k) {
    int n = arr.length;
    HashMap<int, int> hm;
    // Insert first k elements.
    for(i=0; i<k; i++) {          → O(k)
        if (hm.search(arr[i]) == true){
            hm[arr[i]] += 1
        } else {
            hm.insert(arr[i], 1)
        }
    }
    print(hm.size());
    s=1, e=k
    while (e<n)                   → (N-K).
        // sub [s e]
        hm[arr[s-1]] -= 1
        if (hm[arr[s-1]] == 0) {
            // remove key arr[s-1]
            hm.remove(arr[s-1])
        }
```

```
if (hm.search(arr[e]) == true){
    hm[arr[e]] += 1
} else {
    hm.insert(arr[e], 1)
}
s++;
e++;
print(hm.size());
```

3

3

3

(Ques) N 2d points, calculate no. of distinct points.

x[5] = { 2, 3, 2, 5, 3 }
y[5] = { 3, 6, 3, 7, 6 }

3 2 hm = 3

2,3
2,3
hm < ___ >.

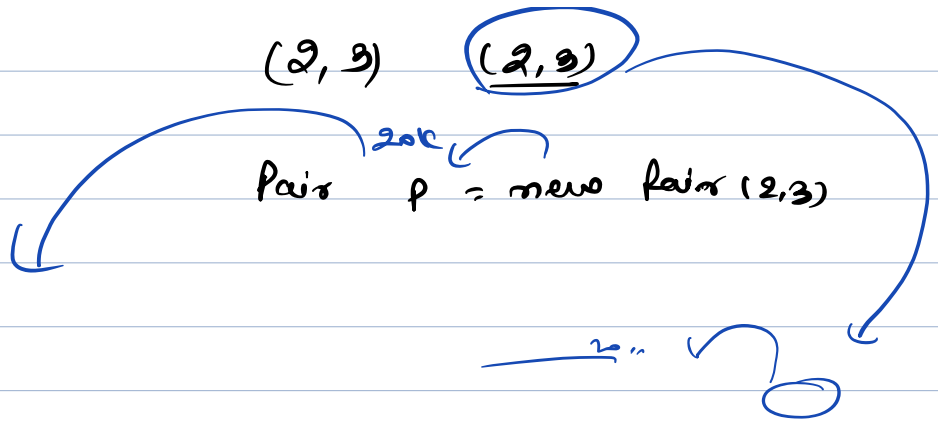
hset <string>

x + @ + y.

(5,7)
(3,6)
(2,3)

2@3 3@6
5@7 3@7.
3@2

2 3 3 6
5 7
2 3 2 3



123 → 6

234 → 9

33 → 6