Facebook
Google

2 - Sum problem

Given an array & a no. K.

Return true if there exists a pair (i, j) in the array

indices

Such that

$$a[i] + a[j] = K$$

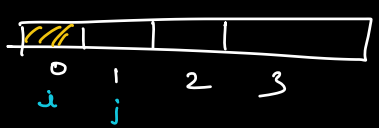
$$i \; != \; j$$

0	1	2	3	4
2,	7,	11,	15,	7

K = 18 → (1, 2) ⇒ True

K = 14 → (1, 4) ⇒ True

K = 20 → ⇒ False



(0,0), (0,1) (0,2) (0,3)

(1,0), (1,1) (1,2) (1,3)

(2,0), (2,1) (2,2) (2,3)

(3,0), (3,1) (3,2) (3,3)

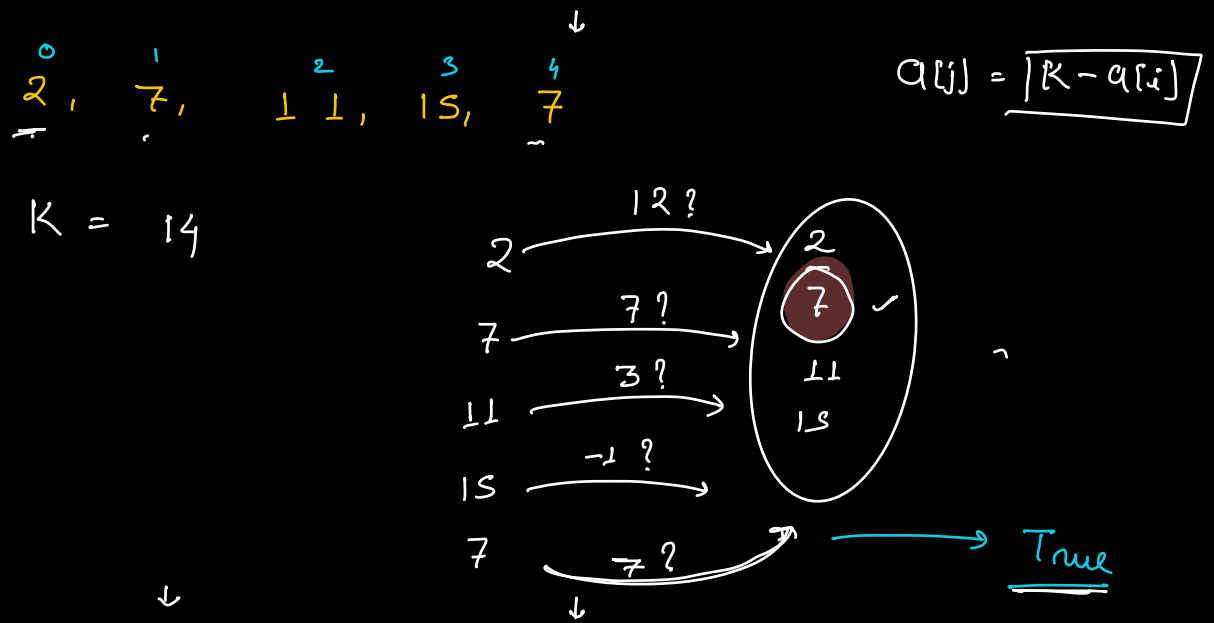
```
for (i=0; i<N ; i++){
    for (j=i+1; j<N; j++){
        if ( a[i] + a[j] == K)
            return true;
    }
```

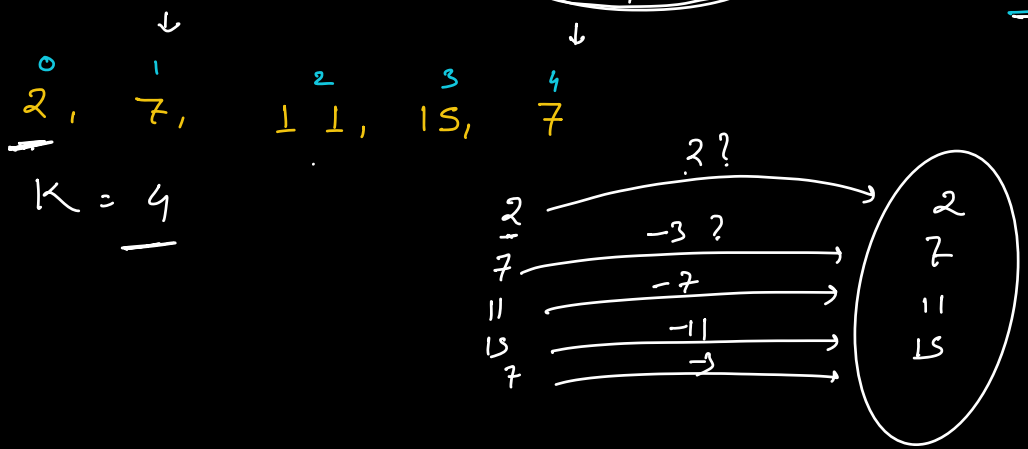
TC: $O(N^2)$

$$a[i] + a[j] = K$$

$$\Rightarrow a[j] = K - a[i]$$

Check if a[j] is present





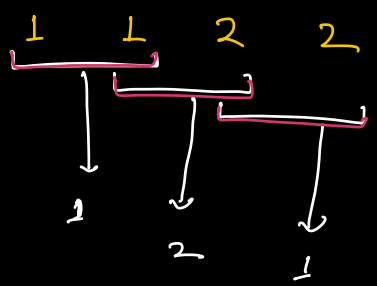
Step I Check if $K - a[i]$ is present in set

Step II insert $a[i]$ to set

What if => Think 30 min

Q Given N array elements. Calculate the no. of distinct elements in every window of size K.

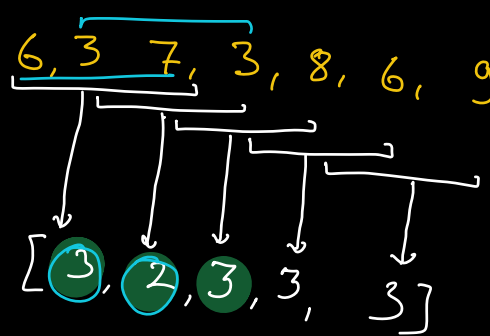
A : 1 1 2 2 K = 2



return : [1, 2, 1]

Quiz 1

A : 6, 3, 7, 3, 8, 6, 9 K = 3



3, 7

=> For every window of size K => N - K + 1

- Add all K elements to an empty set => K
- Add size of the set to ans array.

steps = K (N-K+1)

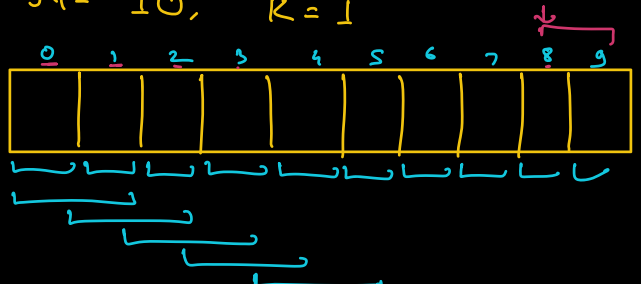
K = N/2

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

$\frac{N}{2}(\frac{N}{2}+1)$

$O(N^2)$

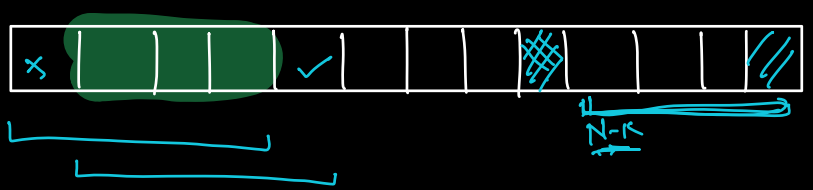
Quiz N = 10, K = 1

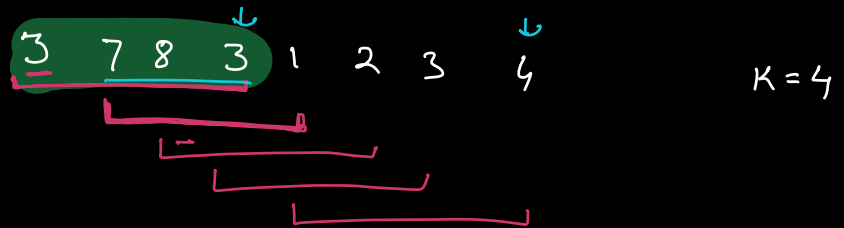


=> N => 10

(N)

K	Start of last window	# of windows
1	0, N-1	N
2	N-2	N-1
3	N-3	N-2
⋮	⋮	N-3
K	N-K	N-K+1





Map

3 : ~~2~~ ~~1~~ 1

4 : 1

1 : 1

2 : 1

3, 4, 4, 3, 4

(I) Process the first window of size K
⇒ Build the freq map

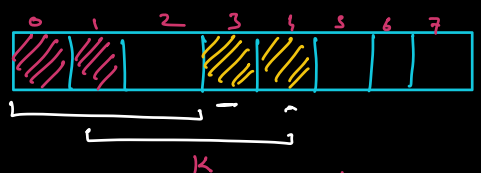
(II) Iterate over remaining windows
⇒ Remove 1st element of prev window
⇒ Add new element.

// Build the freq map

```
for (i=0; i<K; i++) {
    if (map.contains(A[i])) {
        map(A[i])++;
    }
    else
        map.put(A[i], 1);
}
```

⇒ # iteration = K

// Sliding window




```
for (i=0; i<N-K; i++) {
    // Remove A[i]
    Map(A[i])--;
    if (Map(A[i])==0)
        Map.remove(A[i]);

    // Add A[i+K]
    if (Map.contains(A[i+K])) {
        Map.(A[i+K])++;
    else
        Map.put(A[i+K], 1);
}
```

K = 3

Remove index		Add index
0	⟷ K ⟷	3
1	⟷ K ⟷	4
2	⟷ K ⟷	5
:		
i	⟷	i+K

iteration = N-K

Total # of iterations = K + N-K ⇒ N ⇒ O(N)

SC: O(N)

Google Dietic

Q Given an array. Find the length of **largest sequence** which can be rearranged to form a sq of consecutive no.

A: 100, (4), 200, (1), (3), (2)

4, 1, 3, 2 ⇒ 1, 2, 3, 4

⇒ 4.

Quiz -1, 8, 2, 7, 1, 4, 9, 3

1, 2, 3, 4 ⇒ 4

7, 8, 9

Quiz 5, 9, 100, 1, -1, 2, 3, 99, 98, 11, 101, 15, 102

1, 2, 3

100, 99, 98, 101, 102 ⇒ 5

-1, 8, 2, 7, 1, 4, 9, 3

-1, 1, 2, 3, 4, 7, 8, 9

1 2 3 4 1 2 3

Man ✗ ✗ ✗ 4

3, 100, 99, 4, 100, 3, 2, 101, 102

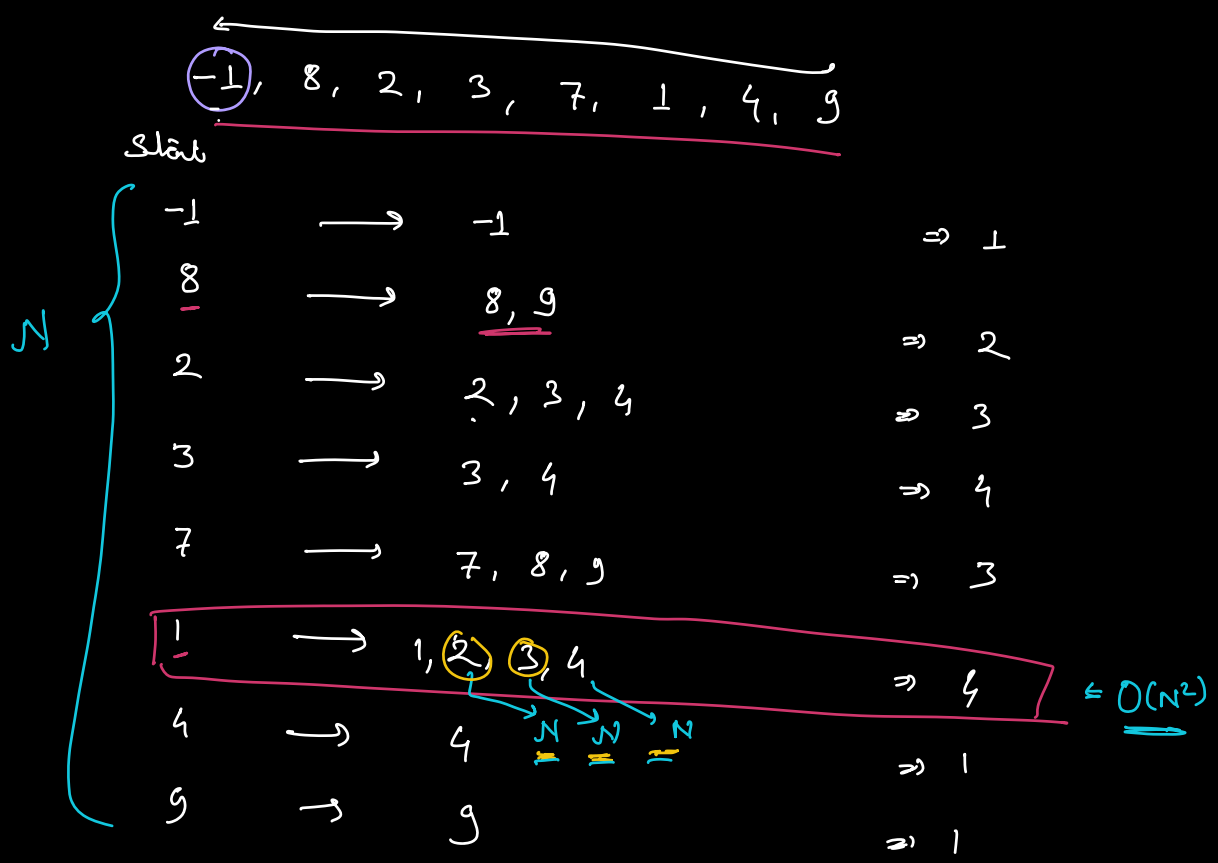
2, 3, 3, 4, 99, 100, 100, 101, 102

C 1 2 2 3 1 2 2 3 4

Max ~~1~~ ~~2~~ ~~3~~ 4

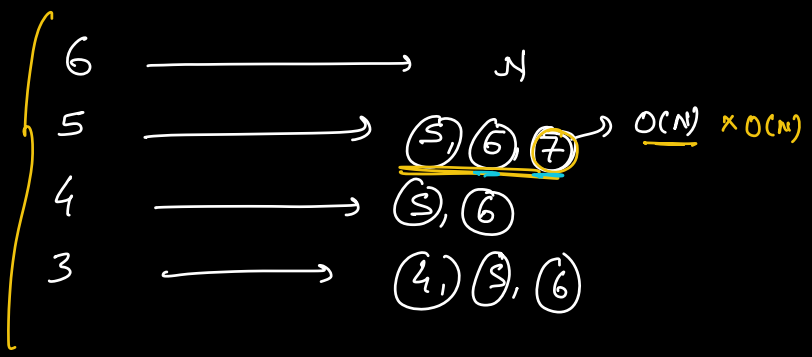
TC: $O(N \log N) + O(N) \Rightarrow O(N \log N)$

SC (Extra): Depends on sorting algo



TC: $O(N^3)$

6, 5, 4, 3, 2, 1




```
// Build the set
Set = { }
for (i=0; i<N; i++) {
    Set.add(a[i]);
}
```

O(N)

// Count seq length for every element as a start

cans = 0;

~~for (i=0; i<N; i++){~~ for every a[i] in Set:

```
    length = 1;  K = 1;
    if (!Set.contains(a[i]-1)) {
        While ( Set.contains(a[i] + K)) {
            length++;
            K++;
        }
        cans = max(length, cans);
    }
}
```

1, 2, 2, 3, 4, 5, 5 6, 7, 8, 9, 10, 11

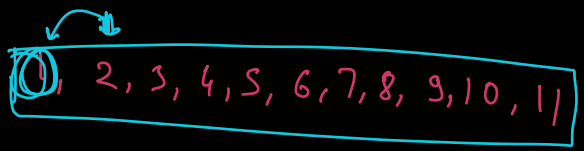
⇒ (1) ⇒ 2, 3, 4, 5, 6, (7) 8, 9, 10, 11, 12 ⇒ O(N).

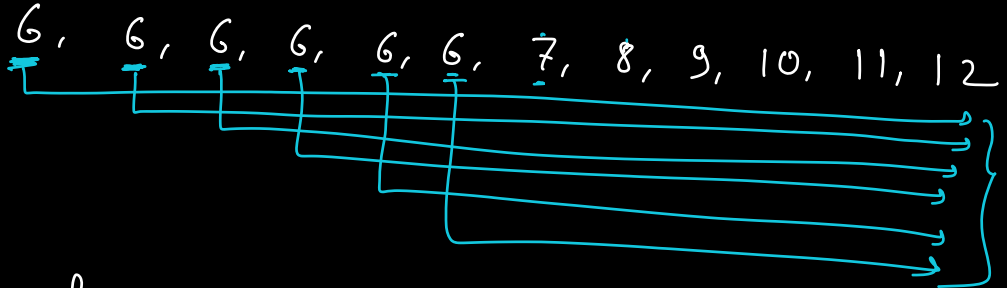
⇒ 2 ⇒ 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 ⇒ O(N).

2 ⇒ 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 ⇒ O(N).

:

} O(N²)





(I) for any a[i] -

✓ if a[i] - 1 is present.

⇒ Stop checking the length of seq starting from a[i]

(II) Iterate over set (Not array)

to avoid duplicate start point.

x 6, 6, 6, 6, 6, 6, 7, 8, 9, 10, 11, 12

9 → is 8 present ? → O(1)

10 → is 9 present ? O(1)

11 → is 10 present ? O(1)

8 → is 7 present ? O(1)

7 →

6 → is 5 present ? x

9, 10, 11
8, 7, 6
12

7, 8, 9, 10, 11, 12, ~~13~~

12 → is 11 present ?

# of iterations = 2N

↑ Because every element being visited at most 2 times.

TC : O(N)

SC : O(N)

TC : N^3 → N^2 → N

SC : 1 → N → N