Fb | Google | so on . . .

Q1. 2 Sum

Given an array & a no. K. Return true if there exists a pair (i,j) in array s.t 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 1 2 3 4 5 6 7 8 9
{8, 9, 1, -2, 4, 5, 11, -6, 7, 5}

K = 6 → True.

{ 0 1 2 3 4 5
{7, 1, 10, -3, 9, 7}

K = -2 ✓

K = 10 ✓

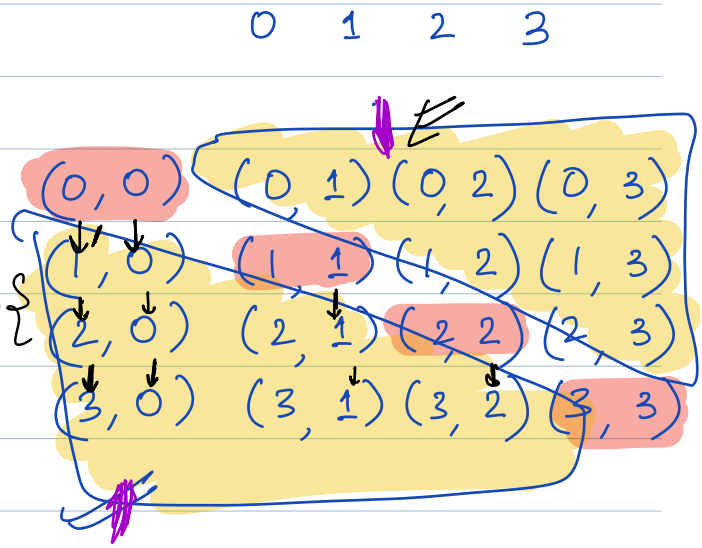
K = 18 X

K = 14 ✓

Brute Force

Create all the pairs.

```
for ( int i = 1; i < N; i++) {
    for( int j = 0; j < i; j++) {
        if ( i != j && A[i] + A[j] == K) {
            return true
        }
    }
}
return false
```



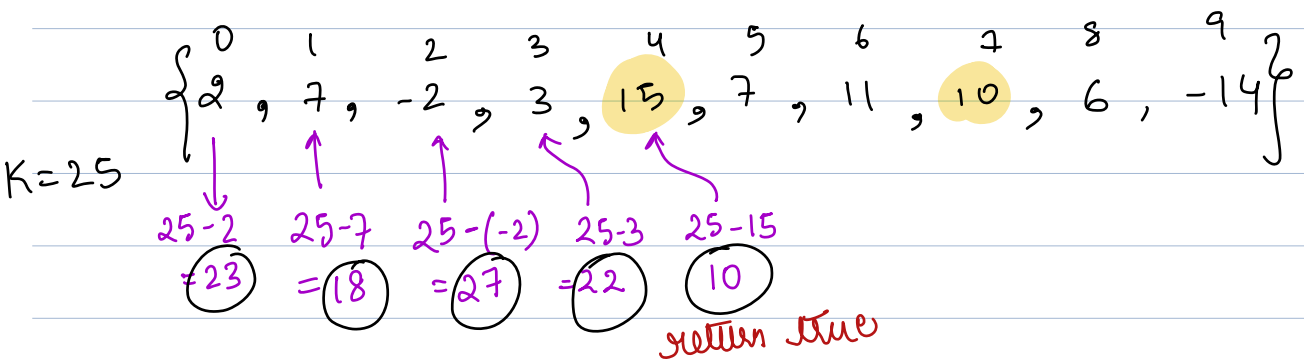
TC → $O(N^2)$

SC → $O(1)$

Observation

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

$A[j] = K - A[i]$



Check whether a value exists or not

↳ ~~Hashmap~~ / HashSet

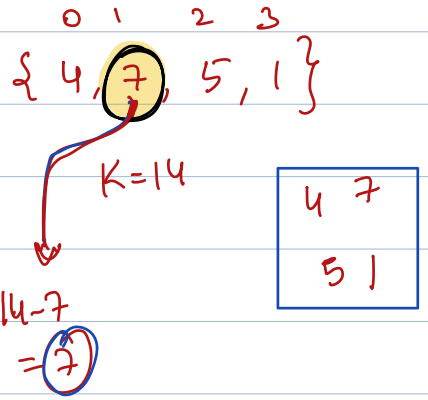
(unique elements)

HashSet s

2	7	-2
3	15	6
11	10	
-4		

```
HashSet<int> s;
for(i=0; i<N; i++){
    s.insert(A[i])
}
for(i=0; i<N; i++){
    x = K - A[i]
    if (x exists in s){
        return true
    }
}
return false
```

Fails for this case



Hashmap → store frequency.

0	1	2	3	4	5	6	7	8	9
2	7	-2	3	15	7	11	10	6	-14

4-2 = 2

K = 4

Hashmap<int, int> mp.

2	1
7	2
-2	1
3	1
15	1
11	1
10	1
6	1
-14	1

```
Hashmap<int, int> mp ;
for(i=0; i<N; i++){
    if (A[i] exists in mp ){
        update (A[i], mp[A[i]]++)
    }
    else {
        insert (A[i], 1)
    }
}
```

frequency

O(N)

```
for( i=0; i<N ; i++){
    x = K - A[i]
    if( x == A[i] && mp[x] >= 2 ) {
```

return true

}

else if (x != A[i] && x exists in mp) {

return true

}

}

return false

O(N)

TC → O(N) + O(N)
= O(N)

SC → O(N)

{ 4, 3, 1, 5 }
0 1 2 3

K = 8

4	1
3	1
1	1
5	1

8 - 4 = 4

Single iteration using set.

K = 25

{ 2, 7, -2, 3, 15, 7, 11, 10, 6, -4 }
0 1 2 3 4 5 6 7 8 9

25-2 = 23
25-7 = 18
25-3 = 22
25-15
25-7 = 18
25-11 = 14
25-10

set s.

2	7
-2	3

25 - (-2) = 10

27

= 15

return true

15	11

K = 12 { 4, 7, 5, 1, 7 }

0 1 2 3 4

12 - 4 = 8

12 - 5 = 7 (true)

12 - 7 = 5

set s

4	7

TC → O(N)

SC → O(N)

```
Hashset <int> s;
for (i = 0; i < N; i++){
    x = K - A[i]
    if (x is is present in s ){
        return true
    }
    else {
        insert (A[i] into the s )
    }
}
```

K = 14

{ 7, 14, 14, 7 }

0 1 2 3

14 - 7 = 7

14 - 14 = 0

14 - 7 = 7

set (unique)

7	14

14 - 14 = 0

K = 14

{ 7, 14, 14, 0, 7 }

14 - 0 = 14

K = 14

7	14

Break → 9 : 13.

A = [7, 14]

0 1

K = 14

set s

7

14

T → false

14-7 = 7   14-14 = 0

A = { 7, 14, 5, 6, 9 } (indices 0, 1, 2, 3, 4)

K = 12 → true

K = 18

{ 7, 14, 0 } (indices 0, 1, 2)   K = 14 → True

14-7 = 7   14-14 = 0   14-0 = 14

7   (14)

14 + 0 = 14

Que. Given N array ele, calc the no. of distinct ele in every window of size K.

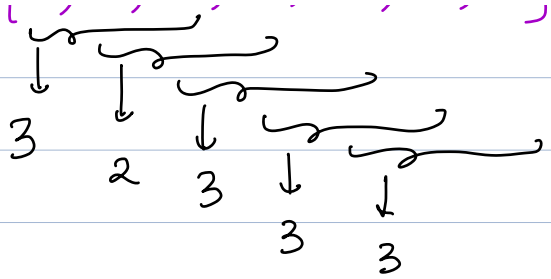
A : { 1 1 2 2 }   K = 2

1   2   1

return { 1, 2, 1 }

QUIZ

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

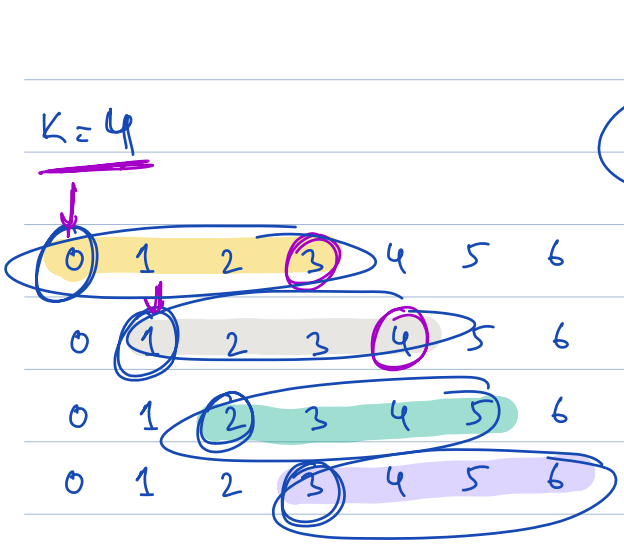


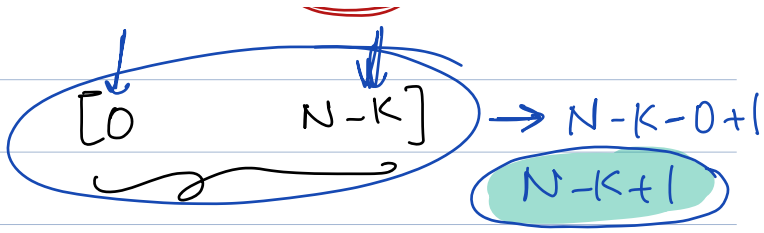
Return {3, 2, 3, 3, 3}

N

K=1	N
K=2	N-1
K=3	N-2
⋮	
K	N-K+1

K=3	start of first window	start of last window
1	0	N-1
2	0	N-2
3	0	N-3
4	0	N-4
⋮	⋮	⋮
K	0	N-K





No. of start points of window of size K = Total no. of windows of size K.

= N-K+1

Brute Force

Iterate over all windows of size K

```
List<int> ans ;
for (i = 0 ; i <= N - K ; i++) {
    j = i+K-1
    HashSet<int> s ;
    for ( z = i ; z <= j ; z++ ) {
        s.insert (A[i])
    }
    ans.push ( s.size)
}
```

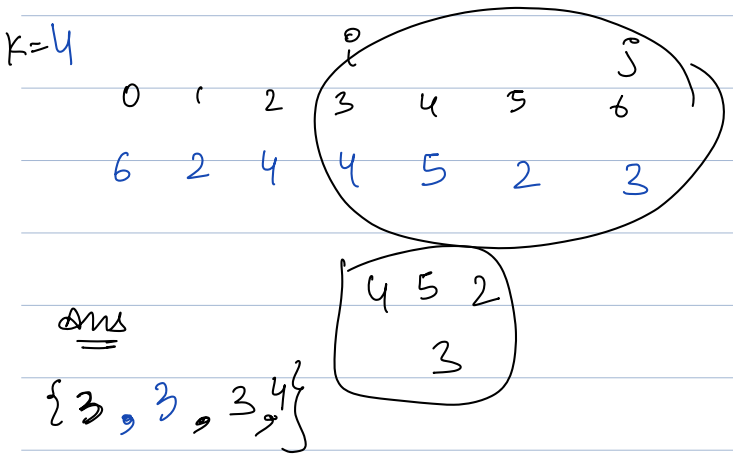
Iterations

(N-K+1) (K)

K=1

(N-1+1)(1) = N

K=N



ans

{3, 3, 3, 4}

$(N - N + 1)(N) = N$

$K \leq N/2$

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

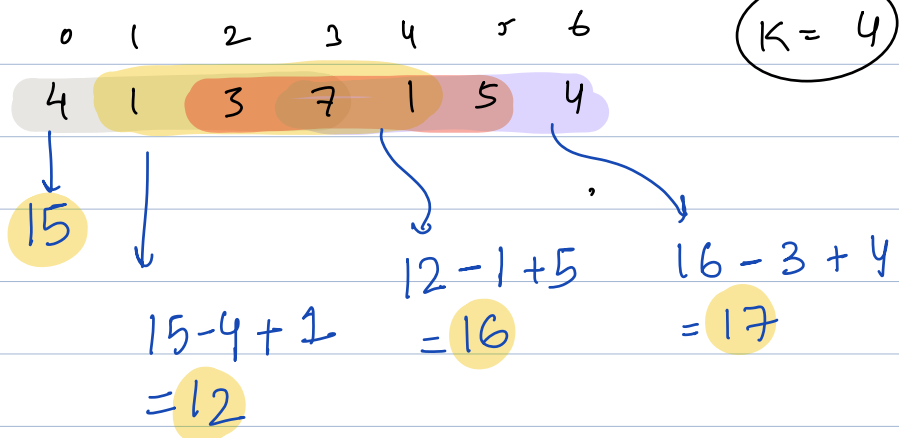
$\left(\frac{N}{2} + 1\right)\left(\frac{N}{2}\right)$

N^2

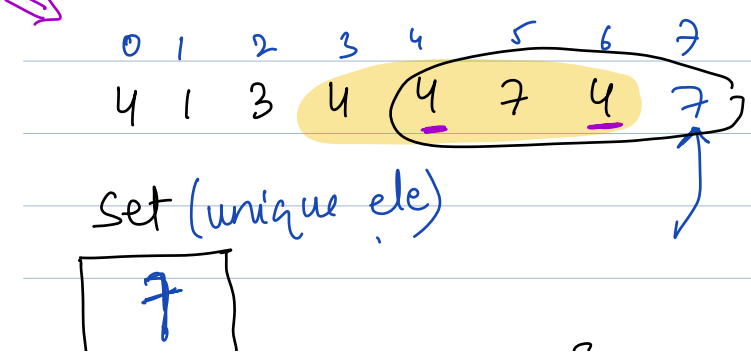
TC → $O(N^2)$

SC → $O(N)$

Sliding window



Can we apply same logic in this question?



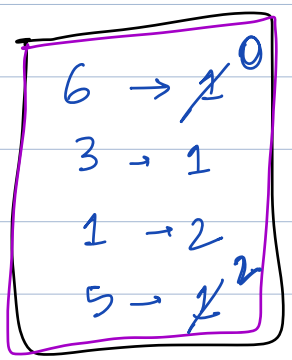
{3, 3, 3, 2, 1}

set !

Hashmap with frequency.

0	1	2	3	4	5	6	7	8	9	10	11
6	3	1	1	5	5	5	6	9	14	9	9

K = 5



{4,

sliding window →

① Process window of size K

⇒ Create the freq map

② Process the rest of the window

⇒ We will add last ele of next window to mp

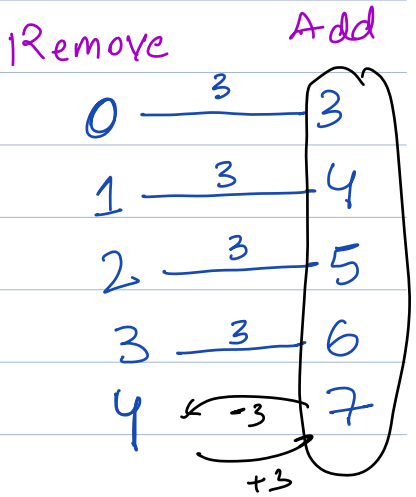
& remove first ele of prev window from map.

```
Hashmap<int> mp;
for(i=0 ; i<K ; i++) {
    if (A[i] is present) {
        mp[A[i]]++
    }
    else {
        insert(A[i], 1)        → frequency
    }
}
list.push(mp.size())
```

Iteration K

```
for(i=0 ; i<=n-K-1 ; i++) {
    x = i+K
    mp[A[i]]--
    if(mp[A[i]]==0){
        remove(A[i] from mp)
    }
    else if(A[x] is present in mp){
        mp[A[x]]++
    }
}
```

K=3



Iteration: N-K

else insert (A[x], 1)

list.push (mp.size())

return list

Total : ~~K + N - K~~ N

O(N)

K=4