Hashing 2 : Problems

Question 1

Given an integer array, check if there exists a pair (i, j) s.t. $\boxed{A[i] + A[j] = K \quad \& \quad i != j}$

eg $A = \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \\ 8 & 9 & 1 & -2 & 5 & 4 \end{array}$

K = 7 ans = true (1,3)

K = 10 ans = true (1,2)

K = 11 ans = false

$A = \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \\ 3 & 5 & 1 & 2 & 1 & 2 \end{array}$

K = 7

ans = true (1,3)

K = 10

ans = false

Bruteforce

```
for ( i = 0 to n-1 ) {
    for ( j = i+1 to n-1 ) {
        if ( a[i] + a[j] == K ) {
            return true;
        }
    }
}
return false
```

(i != j) => use either i > j or i < j

j = i+1

TC = O(N²)

SC = O(1)

Solution → $A[i] + A[j] = K \Rightarrow A[j] = K - A[i]$

$\forall i$, check if there is an element $K - A[i]$

- linear search → $TC = O(N^2)$
- binary search ← sorted data $TC = O(N \log N)$
- Hashset / Hashmap

Hashset	Hashmap
only store single unique element.	<key, value> key ↓ unique

$TC = O(1)$ for all operations

insert/update/delete/search

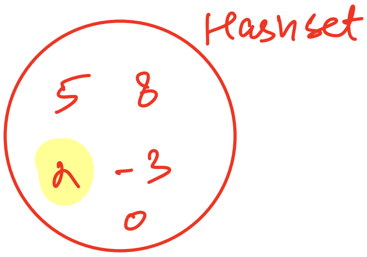
1. Store all elements of A[] in hashset ← $TC = O(N)$
2. $\forall i$, check if $K - A[i]$ is present ← $TC = O(N)$

total $TC = O(N)$

$SC = O(N)$

$A = [5 \quad 8 \quad 2 \quad -3 \quad 0]$ $K = 4$

$K - a[i] =$	$4-5 = -1$ ✗	$4-8 = -4$ ✗	$4-2 = 2$ ✓ $(i == j)$

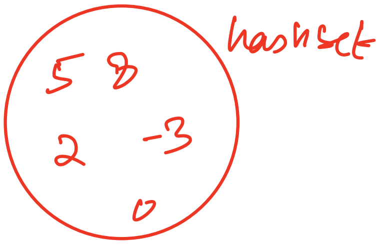


$$\boxed{\begin{array}{c} A[j] = K - a[i] \\ j < i \end{array}}$$

```
for ( i = 0 to n-1) {
        if( hs.contains( K - a[i] ) ) {
                return true
        }
        hs.insert( a[i] )
}
return false
```

A = [5   8   2   -3   0]   K = 4

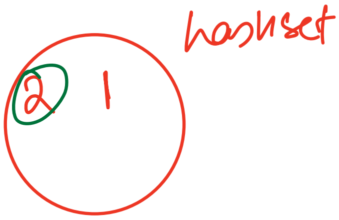
K - a[i] = 4-5 =-1 ✗,   4-8 = -4 ✗,   4-2 =2 ✗,   4-(-3) =7 ✗,   4-0 =4 ✗



ans = false

A = [2   1   3]   K = 5

5-2 =3 ✗,   5-1 =4 ✗,   5-3 =2 ✓



ans = true

Question → Given an integer array, count the number of pairs (i,j) s.t. $\boxed{A(i) + A(j) = K \ \& \ i \,!= j}$

$$\begin{array}{cccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & \\ A = [& 3 & 5 & 1 & 2 & 1 & 2 &] \end{array}$$

(i,j)

K=3

(2,3)
(2,5)
(3,4)
(4,5)

ans=4

```
for (i=0 to n-1) {
    if( hs.contains(K-a[i])){  ←  check if K-a(i) is present
        return true
    }
    hs.insert(a[i])
}
return false
```

check if K-a(i) is present

count # times (K-a(i)) is present

Hashset → Hashmap

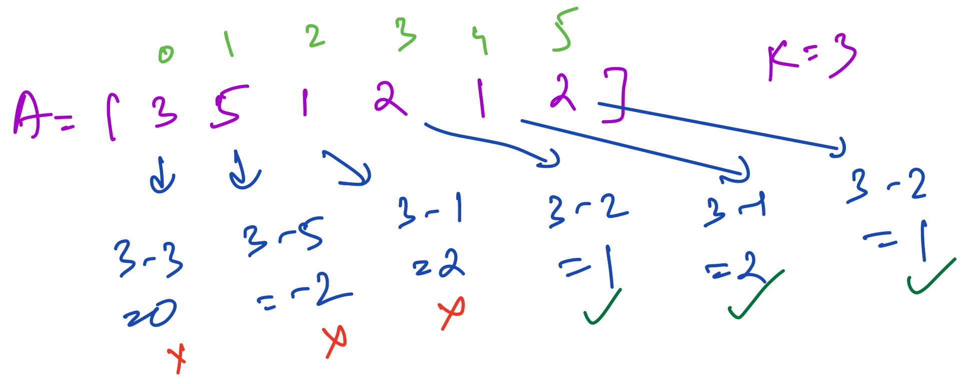
<A(i)>   <A(i), freq of A(i)>

```
ans=0

for (i=0 to n-1) {        // hm → hashmap

    if( hm.contains(K-a[i])){
        ans += hm[K-a[i]];
    }
    if(hm.contains(a[i])){
        hm[a[i]]++
    }
```

TC = O(N)

SC = O(N)

```
        else {
            hm.insert( a[i], 1);
        }
    }
    return ans
```



hashmap

<3,1>

<5,1>

<1,~~1~~ 2>

<2,~~1~~ 2>

$ans = \cancel{0} \xrightarrow{+1} \cancel{1} \xrightarrow{+1} \cancel{2} \xrightarrow{+2} 4$

Question3 Given an integer array, check if there exists a subarray with sum = k.

$$\begin{array}{ccccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ A = [& 2 & 3 & 9 & -4 & 1 & 5 \;] \end{array}$$

K=10 ans = true (0,3)

K=11 ans = true (0,4)

K=20 ans = false

A = [5 10 20 100 105]

K = 110

ans = false

find (i,j) s.t. $A[i] + A[i+1] + \dots + A[j] = K$

$= pf[j] - pf[i-1]$ (prefix sum)

check if there exists (i,j) s.t. $\underline{pf[j] - pf[i-1] = K}$

$$\begin{cases} pf[j] - pf[i-1] = K & i > 0 \\ pf[j] = K & i = 0 \end{cases}$$

$pf[j] = \underline{K + pf[i-1]}$

we can use hashset

$TC = O(N)$

$SC = O(\underbrace{N}_{\text{prefix array}} + \underbrace{N}_{\text{hashset}}) = O(N)$

↓ modify the a[] only

not allowed

```
pf[0] = a[0]
for (i = 1 to n-1)
    pf[i] = pf[i-1] + a[i]
```

→

```
for (i = 1 to n-1)
    a[i] += a[i-1]

//a[i] → prefix sum till
          index i
```

optimize space for pf[] → carry forward

$pf = [\ \underbrace{\boxed{\qquad\qquad}}_{i}\ \underset{j}{9.}\qquad]$

current ↓

$pf[j] - pf[i-1] = K$ (j > i)

$pf[i-1] = pf[j] - K$

↑ check

```
pf = 0

for ( i = 0 to n-1) {

    pf += a[i]

    if ( pf == K) { return true }

    if ( hs.contains( pf - K )) {
            return true

    }

    hs.insert(pf)

}

return false
```

K - pf

K + pf

pf - K ✓✓

	0	1	2	3	4	5
A = [	2	3	9	-4	1	5]
	↓	↓	↓	↓		
pf =	2	5	14	10		
pf-K =	-6	-3	6	2 ✓		

K=8

hashset: 2 5 14

ans = true

Question 4

Given an integer array A, count the # of distinct elements in every subarray of size K.

↑ sliding window

$$A = \begin{array}{ccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 2 & 1 & 3 & 4 & 2 & 3 \end{array}$$

$K = 4$

3 4 4 3

Bruteforce → $\forall$ subarray of size K, count # of distinct elements

$[0 - K-1)$

end → $[K-1, n-1]$

$\Rightarrow$ # subarrays $= (n-1) - (K-1) + 1$

$= n - K + 1$

insert all elements in hashset

ans = size of hashset

$TC = O((n-K+1) \times K)$

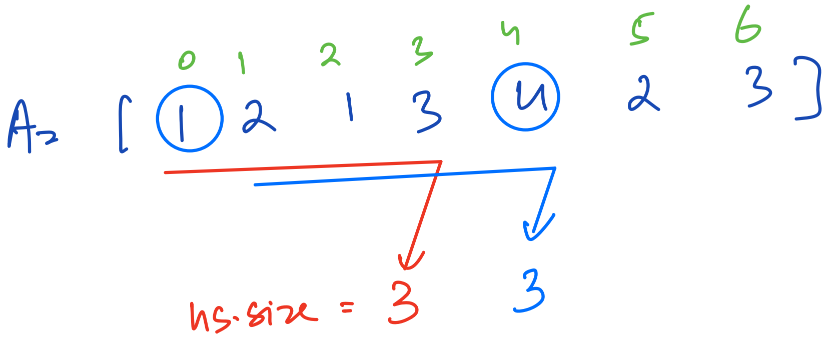
$K = n/2$

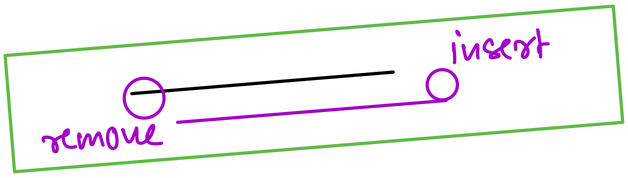
$(n - \frac{n}{2} + 1) \times \frac{n}{2} = \frac{n^2}{4}$

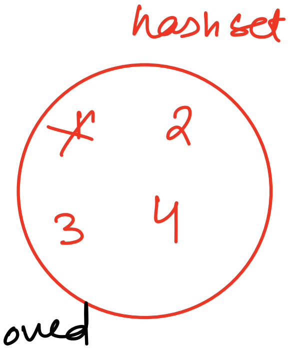
$TC = O(N^2)$

$SC = O(K)$

Sliding window + hashset



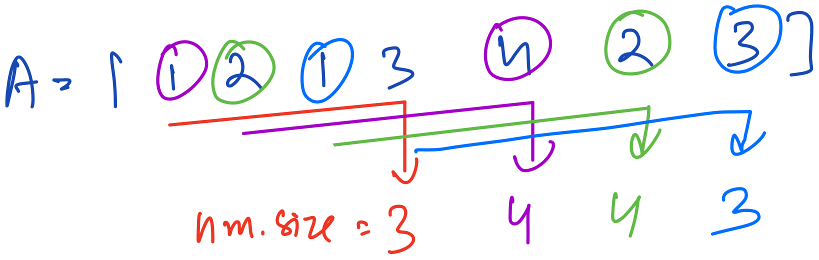




when we removed a[0]=1, it also removed a[2] i.e, all occurances of 1 from hashset

important to store frequency of a[i]

⇒ use hashmap



K=4

hashmap

<~~1,1,2~~> 0 → remove

<2,~~1~~> 0 1

<3,~~1~~> 2

<4,1>

TC = O(N)   SC = O(K)

Code

```
for (i=0 to k-1) {          → for first subarray
    if(hm.contains(a[i])){
        hm[a[i]]++
    }
    else {
        hm.insert(a[i], 1);
    }
}
print (hm.size)

s=1 , e=K          → indices of 2nd subarray

while ( e<n ) {
    // remove a[s-1]
    hm[a[s-1]]--;
    if( hm[a[s-1]] == 0){
        hm.remove(a[s-1])
    }
    // add a[e]
    if(hm.contains(a[e])) {
        hm[a[e]]++
}
else {
    hm.insert(a[e], 1);
}
print( hm.size)

s++ , e++

}
```

TC = O(N)

SC = O(K)

Doubt

P 2 attributes   P p = new C()

↓

C 1 attributes

total = 2+1

$\boxed{\text{C object}}$ 3 attributes

new C()