Agenda

1. Pair with given sum - 2
2. Pair with given difference
3. Subarray with sum K
4. Container with most water

The more you sweat in peace,
the less you bleed in war

Fri 24 May Contest

↓

Sat 25 May Discussion
9 PM

1. Given an integer sorted array A and an integer K, find any pair (i, j) such that A[i] + A[j] = K and i != j

	0	1	2	3	4	5	6	
A =	[-5	-2	1	8	10	12	15]	K = 11

Ans. i j A[i] + A[j]

(2, 4) 1 + 10

YES (2,4)

	0	1	2	3	4	5	6	7	8	9
A =	[-3	0	1	3	6	8	11	14	18	25]

K = 12

i , j A[i] + A[j]

YES , (2, 6) 1 + 11

Approach 1 : Brute Force

Go to all pairs and check for sum = K

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

TC : O(N²)

SC : O(1)

Approach 2 : For every A[i], look for K - A[i] on the right side (i+1 → N-1)

a[i] + a[j] = k
a[j] = k - a[i]

```
for (i=0 ; i<n ; i++) {
    // partner = k - A[i]
    int j = binarysearch(partner, i+1, N-1)
    if (j != -1)    (i, j)
}
return (-1, -1)
```

TC: O(N log N)
SC: O(1)

Approach 3: Hashset (Hashmap
ele : idx
key : value)

① For every element A[i], look for partner (k - A[i]) in the hashset.

② Add cur element A[i] in hashset before going to next idx.

TC: O(N)   SC: O(N)

Approach 4: 2 pointer approach

TC: O(N)
SC: O(1)

	i	j						
	0	1	2	3	4	5	6	
A =	[-5	-2	1	8	10	12	15]	K = 11

$A[i] + A[j] = -5 + -2 = -7 < 11$

Increase i^{th} ele, j^{th} ele or both

$$\begin{array}{lcccccccl} & & & & & & i & j & \\ & & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ A = & [& -5 & -2 & 1 & 8 & 10 & 12 & 15 \ ] \end{array} \qquad K=11$$

$A[i] + A[j] = 27 > 11$

↓ or ↓ or both

$$\begin{array}{lcccccccl} & & & & i & & j & & \\ & & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ A = & [& -5 & -2 & 1 & 8 & 10 & 12 & 15 \ ] \end{array} \qquad K=11$$

i	j	A[i] + A[j]	sum	
0	6	-5 + 15 = 10	10 < 11	Increase sum ↓ i++

(bcoz j can't be ↑)

i	j	A[i] + A[j]	sum	
1	6	-2 + 15 = 13	13 > 11	Decrease sum ↓ j--

already eliminated ← (bcoz i can't be ↓)

i	j	A[i] + A[j]	sum	
1	5	-2+12 = 10	10 < 11	Increase sum ↓ i++
2	5	1+12 = 13	13 > 11	Decrease sum j--

2 4 1 + 10 = 11 11 == 11 break

```
int i=0, j=N-1
while (i<j) {
    if (A[i] + A[j] == k) {
        return (i,j)
    }
    else if (A[i] + A[j] < k) {
        i++
    }
    else
        j--
}
return (-1,-1)
```

TC: O(N)
SC: O(1)
↓
sorted array

---

2. Count all pairs in a sorted array whose sum is K.

A = [1 2 3 4 5 6 8] K = 10

Ans : 2 (2,8) (4,6) element pair

2 pointer Approach

Case 1 : when elements are distinct

```
ans=0
int i=0, j=N-1
while (i<j) {
    if (A[i] + A[j] == k) {
        ans++   i++   j--
    }
    else if (A[i] + A[j] < k) {
        i++
    }
    else
        j--
}
return ans
```

2 3 4 6 8

K=6

TC: O(N)

SC: O(1)

Case 2: when elements are duplicate

(A) using freq array / Hashmap

A = [2 3 3 10 10 10 15]   K=13

↓

A' = [2 3 10 15]

HM

3 : 2
2 : 1
10 : 3
15 : 1

i	j	sum		ans=0
0	3	17	17 > 13	j--
0	2	12	12 < 13	i++
1	2	13	13 == 13	freq[3] × freq[10]

ans = 2×3=6

2 1 i > j Break

ans = 0
int i = 0, j = N-1
while (i < j) {
if (A[i] + A[j] == k) {
ans += freq[A[i]] * freq[A[j]]
i++ j--
}
else if (A[i] + A[j] < k) {
i++
}
else
j--
}

return ans

TC: O(N)

SC: O(N)

↓

Hashmap

(B) Not use freq array / Hashmap

id

A = [2 3 3 10 10 10 15] K = 13

i j

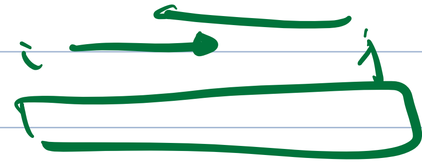
jd

cnt_i = 2 cnt_j = 3

i j

3 3 3 4 5 10 10

id jd



```
ans=0
int i=0, j=N-1
while (i<j) {
    if (A[i] + A[j] == k) {
        if (A[i] != A[j]) {
            int id = i, jd = j, cnti=0, cntj=0
            while (id<n && A[id]==A[i]) {
                cnti++   id++
            }
            while (jd>=0 && A[jd]==A[j]) {
                cntj++   jd--
            }
            ans += cnti * cntj
            i = id    j = jd
        }
        else {  cnti = j-i+1
            ans += (cnti) * (cnti-1)/2        break
        }
    }
    else if (A[i] + A[j] < k) {
        i++
    }
    else
        j--
}
return ans
```

TC: O(N)

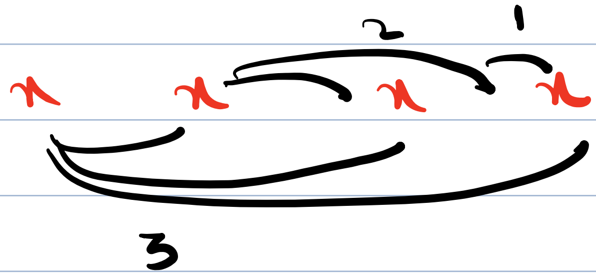
SC: O(1)

1

4 4 4

2

3 pairs



$K = 2x$

$3+2+1=6$ pairs

$4\ x \rightarrow \frac{3 \times 4}{2}$

$N\ x \rightarrow \frac{N \times (N-1)}{2}$

10:36

---

3. Given an integer sorted array A and an integer K, find any pair (i, j) such that $A[j] - A[i] = K$, $i \,!= j$ and $K > 0$.

$$\begin{array}{ccccccccc} & & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ A & = & [-5 & -2 & 1 & 8 & 10 & 12 & 15] \end{array} \quad K = 11$$

$K>0 \Rightarrow A[j] - A[i] > 0 \Rightarrow A[j] > A[i]$
$\Rightarrow j > i$

Ans $(i, j) = (2, 5)$ $12 - 1 = 11$

Approach 1: Brute Force

Go to all pairs and check for diff $= K$

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

TC : O(N²)

SC : O(1)

$$\begin{array}{cccccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 & \\ A = [& -5 & -2 & 1 & 8 & 10 & 12 & 15 &] & K = 11 \end{array}$$

A[i]	A[j]
-5	6
-2	9

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

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

Approach 3 : 2 pointers

$$\begin{array}{cccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & \\ A = [& 1 & 2 & 4 & 5 & 6 & 12 &] & K = 10 \end{array}$$

Ans → (i, j) = (1, 5)

12 − 2 = 10

$$\begin{array}{cccccccccc} & i & & & & & & j & \\ & 0 & 1 & 2 & 3 & 4 & 5 & 6 & \\ A = [& -5 & -2 & 1 & 8 & 10 & 12 & 15 &] & K = 11 \end{array}$$

$A[j] - A[i] = 15 - (-5)$

$= 20 > 11$

Decrease ↓

↓ A[j] or ↑ A[i] or both

b - a = diff

j-- or i++ or both

b - a

$\downarrow 8 - 2 \uparrow = 6 \downarrow$

$5 - 2 = 3$

$8 - 4 = 4$

$-2 - \min = 3 < 11$

		i			j		
	0	1	2	3	4	5	6
A =	[-5	-2	1	8	10	12	15]

k = 11

i	j	diff (A[j]-A[i])		
0	1	-2 - (-5)	3 < 11	b - a = diff ↑b or ↓a

inc j or ~~dec i~~

Increase diff
↓
j++

0	2	1 - (-5)	6 < 11	Increase diff j++
0	3	8 - (-5)	13 > 11	Decrease diff

↓b or ↑a

b - a = diff ↓

~~↓j~~ or ↑i

Increase i

i++

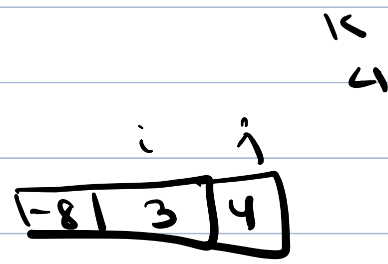
1	3	8 − (−2)	10 < 11	Increase diff j++
1	4	10 − (−2)	12 > 11	Decrease + i++
2	4	10 − 1	9 < 11	Increase diff j++
2	5	12 − 1	11	Stop

```
int i=0 , j=1
while (j<N) {
    if (A[j] - A[i] ==k) {
        return (i, j)
    }
    else if (A[j] -A[i] < k) {
        j++
    }
    else {
        i++
    }
}
return (-1, -1)
```

k>0

TC: O(N)

SC: O(1)



all +ve

4. Given an integer array A and an integer K, check if there exists a subarray with sum K.

A = [1 3 15 10 20 3 23] K=33

K = 33 Ans → True [10, 20, 3]

K = 44 Ans → False

Approach 1 : Use pf [ ]

K=33

A = [1 3 15 10 20 3 23]

pf = [1 4 19 29 49 52 75]

52 − 19 = 33

① sum (i − j) = K

pf [j] − pf [i−1] = K

Array of all +ve

pf [ ] → sorted array

In pf [ ] array, find a pair (i, j) where difference = K

② sum (0 − j) = K ⇒ pf [j] = K

Check every value in pf [ ] with K

TC : O(N) SC : O(N)

SC : O(1) Modify original arr

2 pointers

K = 33

A = [1 3 15 $\boxed{10 \quad 20 \quad 3}$ 23]

i j

sum = 1 → 4 → 19 → 29 → 49

48 → 45 → 30

30 → 33

K = 7

0	1
2	4

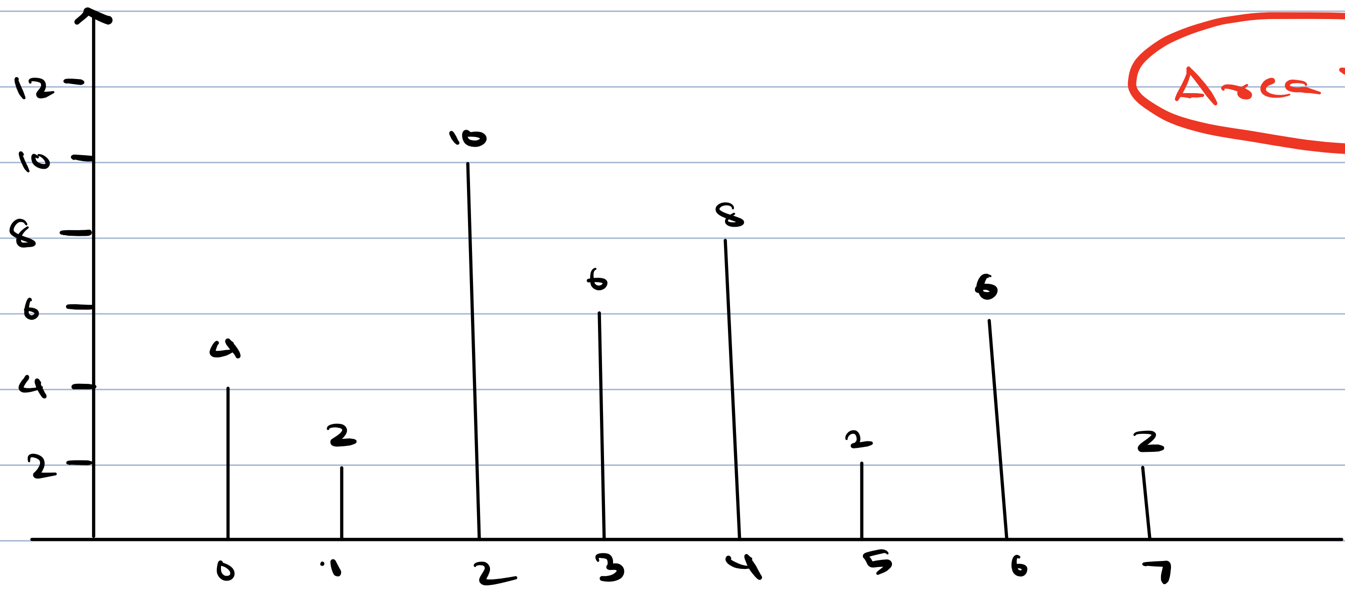
```
i=0 , j=0 , sum= A[0]

while (j < N) {
    if (sum == K)
        return true
    else if (sum < K) {
        j++            if (j==N) break
        sum += A[j]
    }
    else {
        sum -= A[i]
        i++
    }
}
return false
```

TC : O(N)

SC : O(1)

5. Given an integer array A where array element represents height of the wall. Find two walls that can form a container to store maximum water.

A = [4 2 10 6 8 2 6 2]



Area = 24

$A = h \times w$

① water b/w idx 1 and 4

walls 2 and 8

$2 \times 3 = \underline{6}$

② idx 4 and 6

min(8,6) × (6−4)

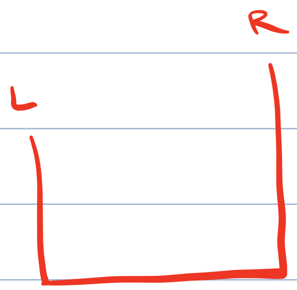
$6 \times 2 = \underline{12}$

③ idx 2 and 6

↓

$\min(10, 6) \times (6-2)$

$6 \times 4 = 24$



water = min(A[L], A[R]) * (R − L)

$$A = \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \end{array}$$
$$A = [\,4 \quad 2 \quad 10 \quad 6 \quad 8 \quad 2 \quad 6 \quad 2\,]$$

i j

ans = ~~14~~ 24

water = Height × width

i	j	h	w	water	
0	7	2	7	14	A[i] > A[j] j--
0	6	4	6	24	A[i] < A[j] i++
1	6	2	5	10	A[i] < A[j] i++

To see if we can get more water, we should try to increase height at cost of same width

2 6 6 4 24 A[i] > A[j]

j --

i = 0 , j = N - 1

ans = 0

while (i < j) {

water = min (A[i], A[j]) * j - i

ans = max (ans, water)

if (A[i] < A[j]) {

i ++

}

else

j --

}

return ans

TC : O(N)

SC : O(1)