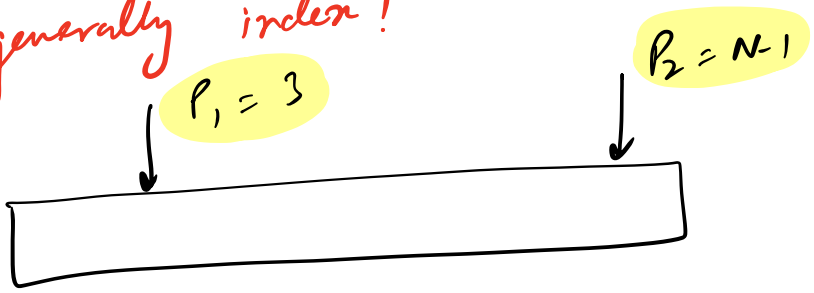
# Two Pointers

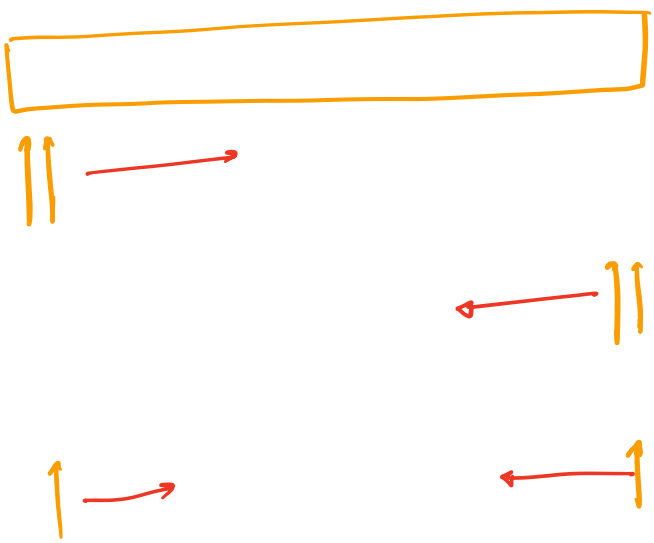
→ Not the c++ pointers!

→ generally index!




① 1) Where to place the 2 ptrs initially?

2) How to move the 2 ptrs? [Deterministic]



# 2-SUM

Q Given a sorted Array A an int. K.
Find a pair (i, j) : $A_i + A_j = K$

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| A: | 1 | 3 | 5 | 10 | 20 | 23 | 30 |

(4, 1), (1, 4) ⟵ K = 23

(4, 4) (6, 3) (3, 6) ⟵ K = 40

I) BF

```
f ( i : 0 ⟶ N-1)
    f ( j : 0 ⟶ N-1)
        if (Ai + Aj == K)
```

TC : O(N²)

SC = O(1)

II) BS

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| A: | 1 | 3 | 5 | 10 | 20 | 23 | 30 |

K = 23

fix

find 23-1 = 22

BS ✓

$A_i + A_j = K$

$A_j = K - A_i$

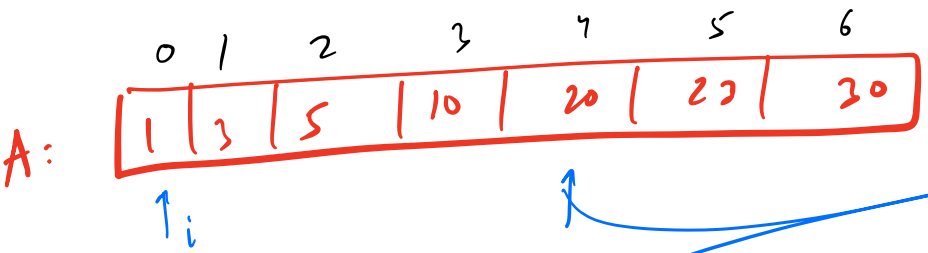
```
for (i: 0 ——→ N-1) {
    f = K - A[i];
    if( BS( A[], i, N-1, f)) {
        ==========
    }
}
```

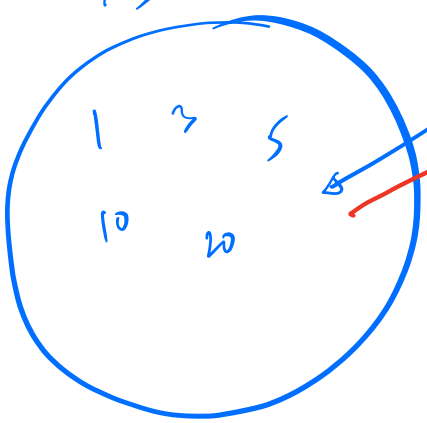
TC = O(N lg N)

SC = O(1)

K = 20

III HS →



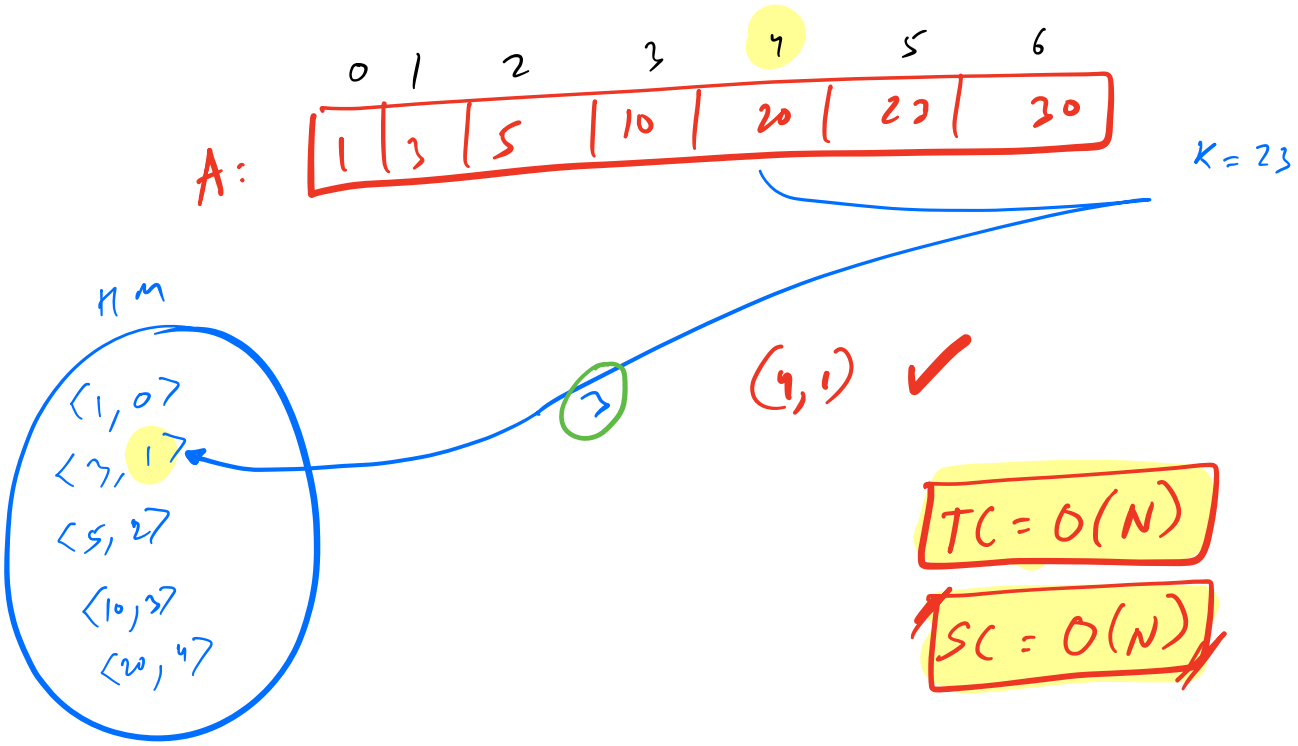




3 ?

this idea can tell us if a pair exists but not the indexes!

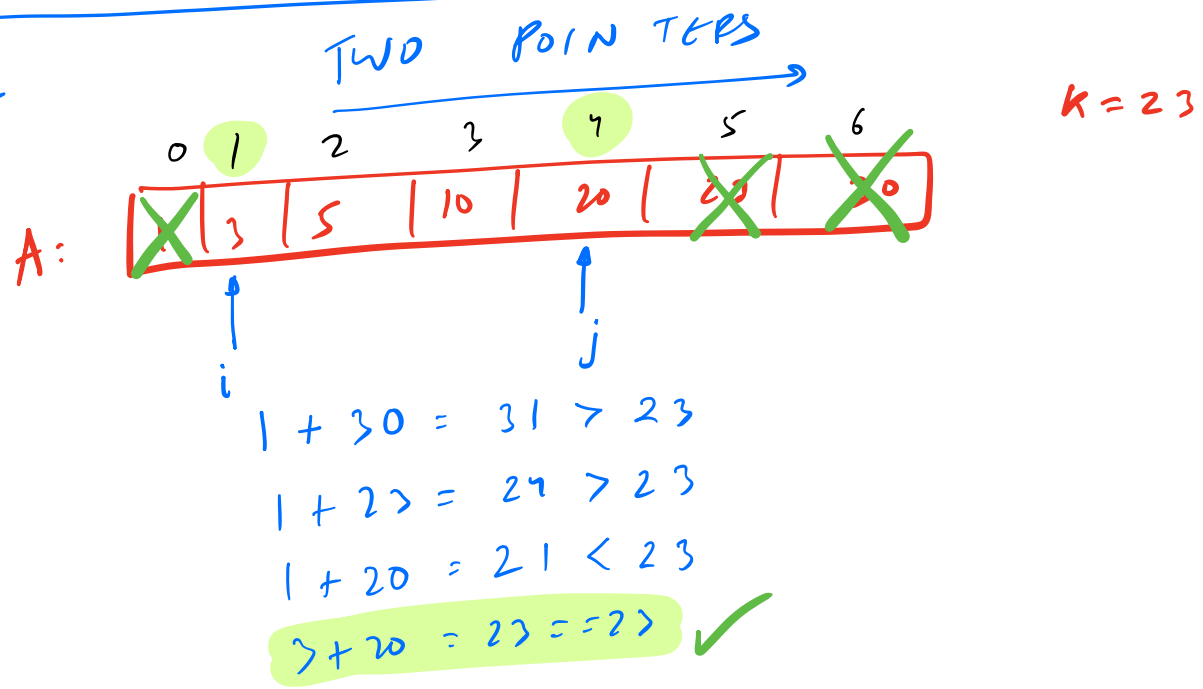
## III) Hash Map

< A[i], i >




---

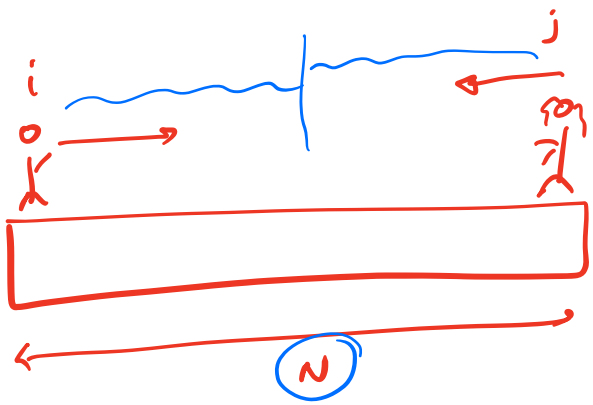
## IV



```
i = 0,  j = N-1;
while ( i <= j ) {
    if ( A[i] + A[j] > K) j--;
    else if ( A[i] + A[j] < K) i++;
    else { ret (i, j); }
}
ret (-1, -1);
```

TC = O(N)

SC = O(1)



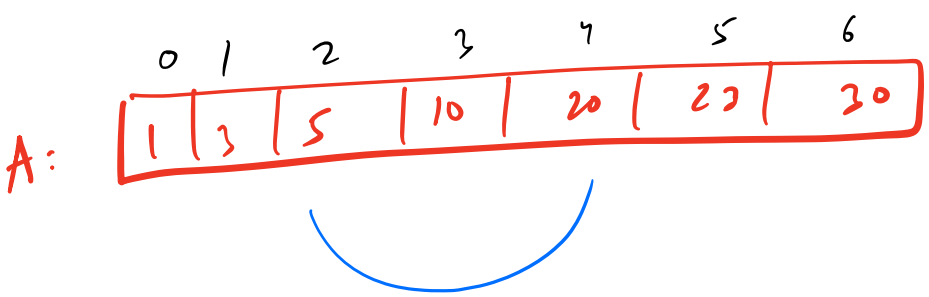

---

Q Given a sorted Array of K.

Find a pair (i, j) : $A_j - A_i = K$

$K \geq 0$

$i <= j$




K = 15

i j

(2, 4)

$A_j - A_i$

20 − 5 = 15 ✓

1) BF ✓

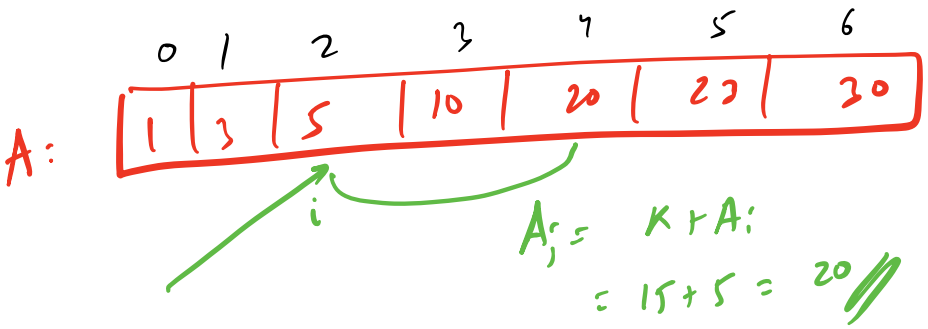
## II) BS ✓

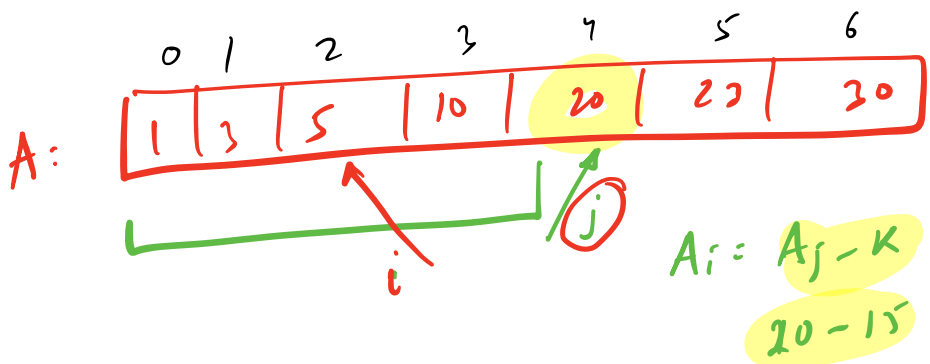
1) $A_j - A_i = K$ (find $A_i$)

$$A_j = K + A_i$$

2) $A_j - A_i = K$ (find $A_j$)

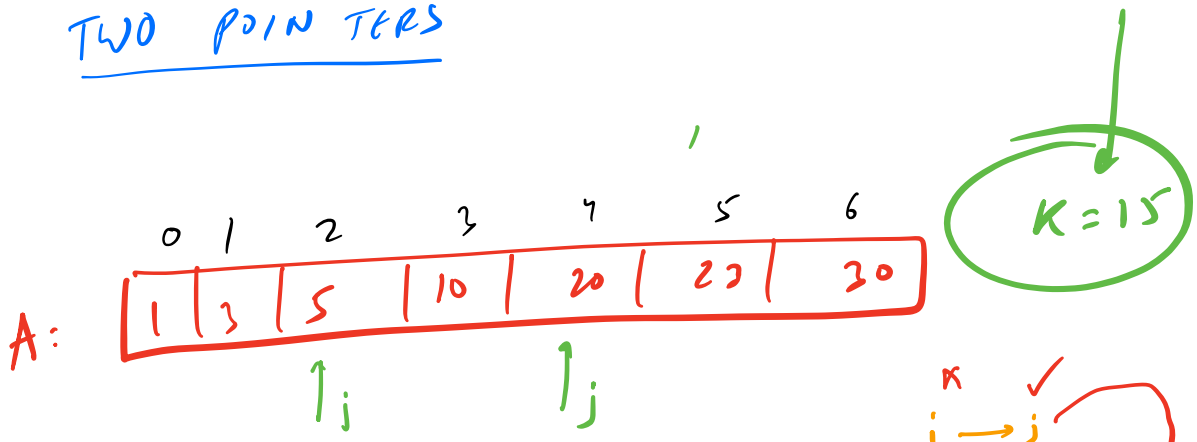
$$A_i = A_j - K$$

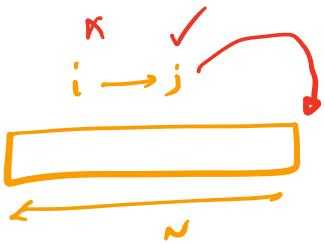



## III) HM ✓

## (IV) TWO POINTERS




$1-1 = 0 < 15$

$3-1 = 2 < 15$

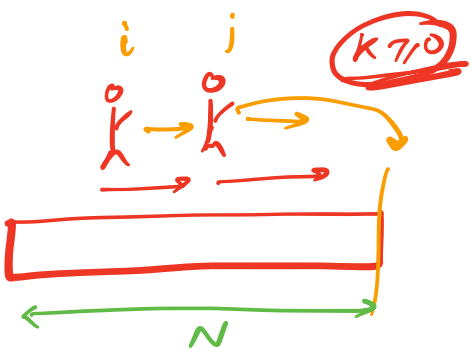
$5-1 = 4 < 15$

$10-1 = 9 < 15$

$20-1 = 19 > 15$

$20-3 = 17 > 15$

$20-5 = 15 == 15$ ✓ (2, 7)




```
i = 0      j = 0;
while( j < N ){
  if( A[j] - A[i] < K) j++;
  else if( A[j] - A[i] > K) i++;
  else { ret (i, j); }
}
ret (-1, -1);
```

TC = O(N)

SC = O(1)

# 3-SUM

Given a sorted Array, K.
Find a triplet (i, j, k):

$$A_i + A_j + A_k = K$$

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| A: | 1 | 3 | 5 | 10 | 20 | 23 | 50 |

K = 38

(2, 3, 5)

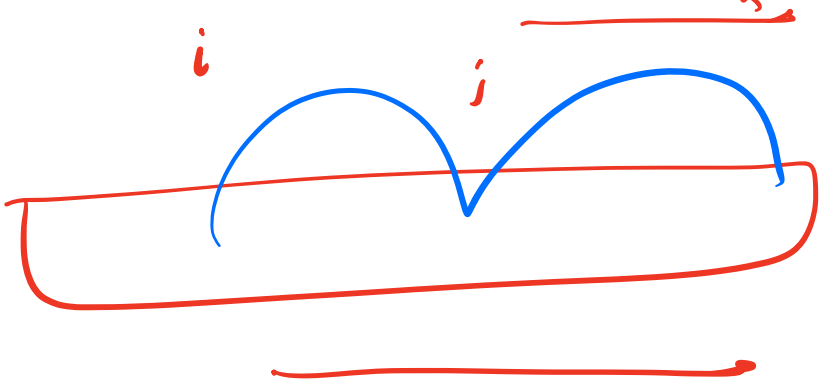
5 + 10 + 23

1) BF : $f(i: 0 \longrightarrow N-1)$

$f(j: i \longrightarrow N-1)$

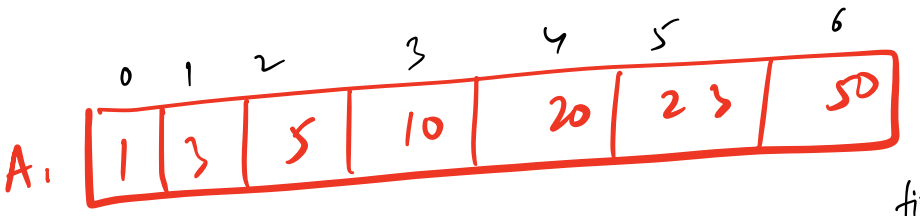
$f(K: j \longrightarrow N-1)$

$N^3$



II) BS

$i < j < K$




$K = 38$

fixing

$A_i + A_j + A_k = K$

$A_k = K - (A_i + A_j)$

```
f( i: 0 ⟶ N-1)
   f( j: i ⟶ N-1)
        f = K-(A[i] + A[j]);
        BS(A[], j, N-1, f);
```

$TC = O(N^2 \log(N))$

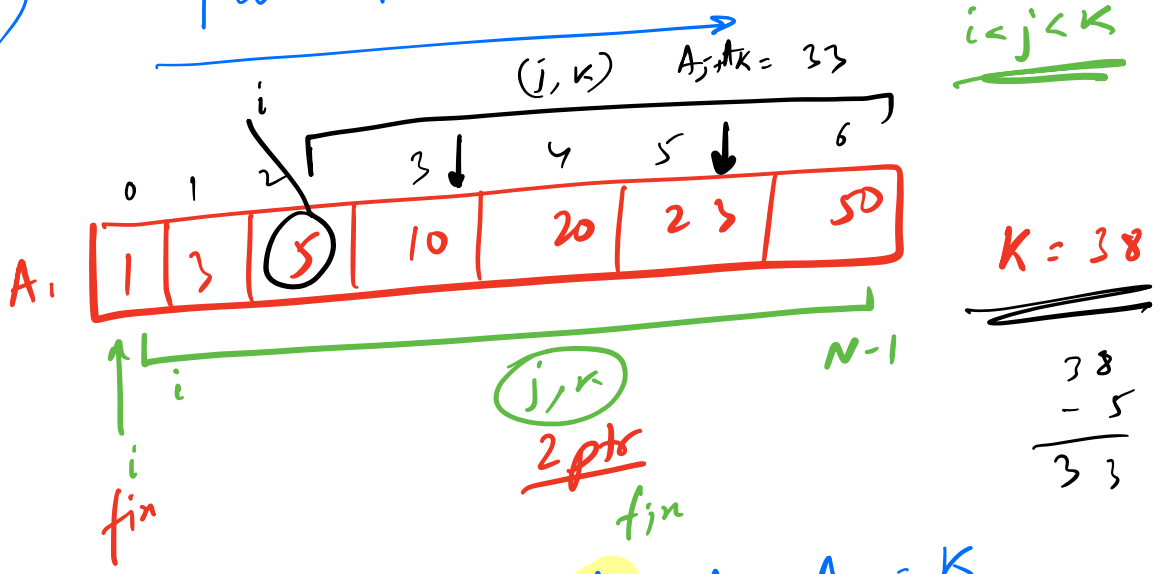
$SC = O(1)$

III MM

$TC = O(N^2)$

$SC = O(N)$

# IV) TWO POINTERS




$A_i + A_j + A_K = K$

$A_j + A_K = K - A_i$

```
f( i = 0 ⟶ N-1) {
    Twoptr( A[], i, N-1, K-A[i])
}
```

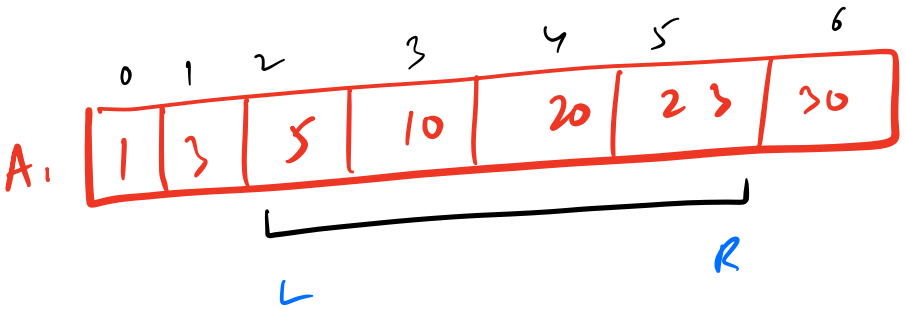
N, N

$TC = O(N^2)$

$SC = O(1)$

Q Given an Array of +ve elements.
Find a sub-array whose sum == k

k >= 0




K = 58

(2, 5)

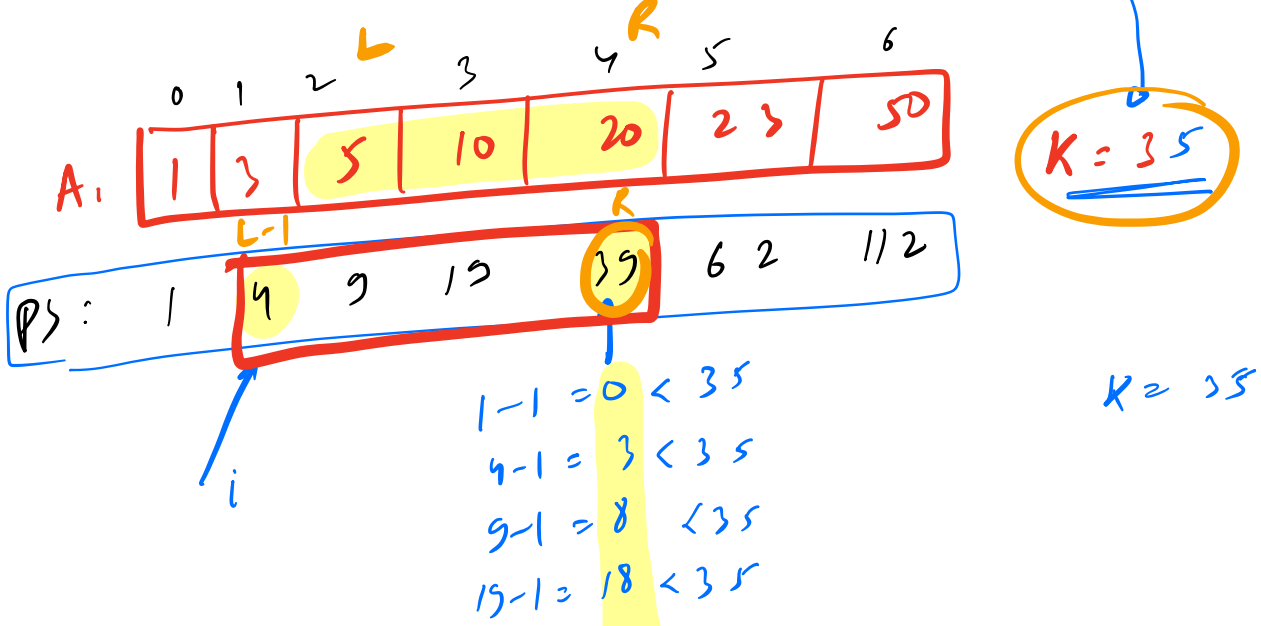
PrefixSum

PS: []

$K = Sum(L, R) = PS[R] - PS[L-1] = k$

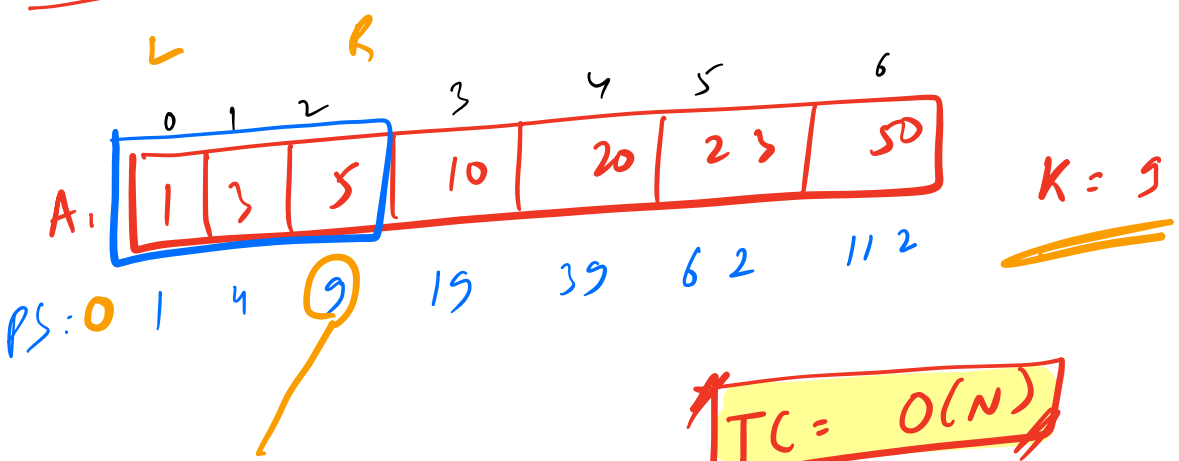
PS[] is SORTED!

find 2 elements in PS[] whose diff == k



39-1 = 38 > 35

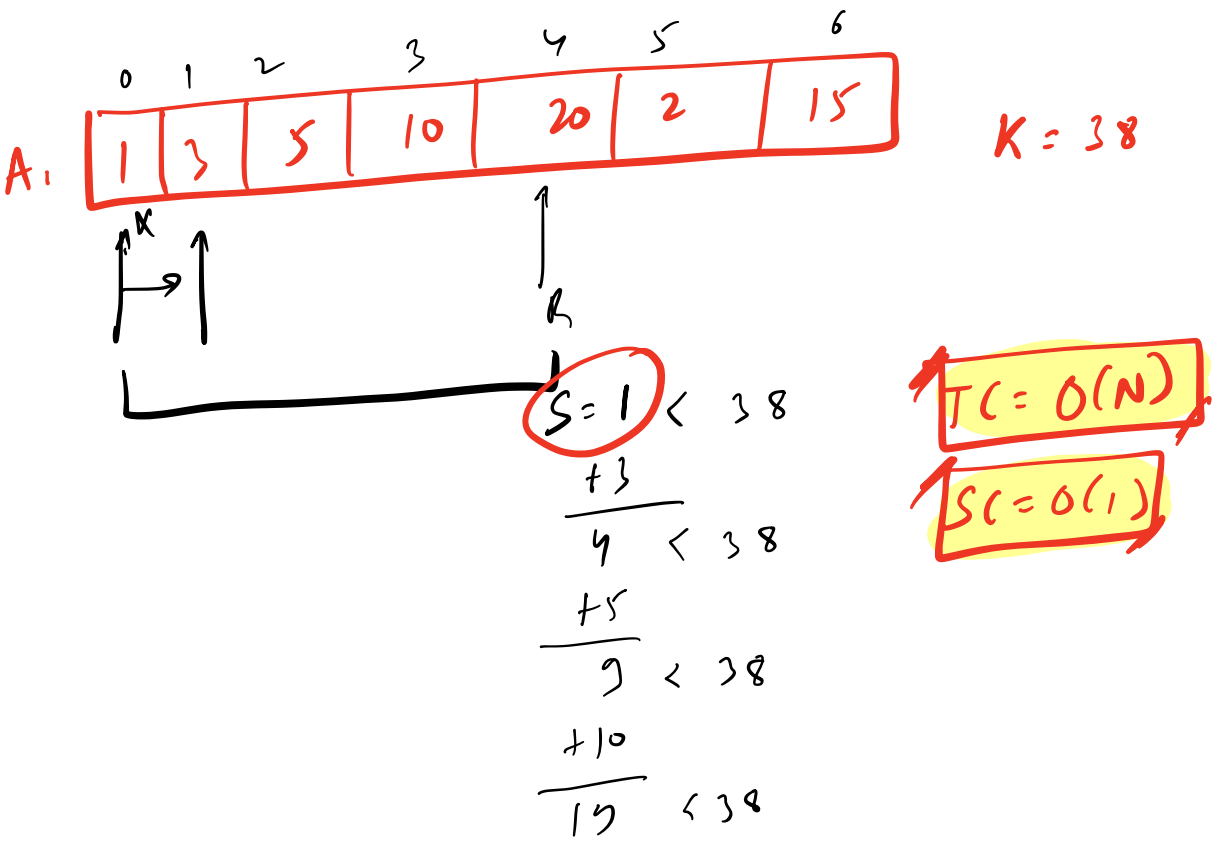
CORNER CASE → 39-4 = 35 = 35




TC = O(N)

SC = O(N)

How to improve?




TC = O(N)

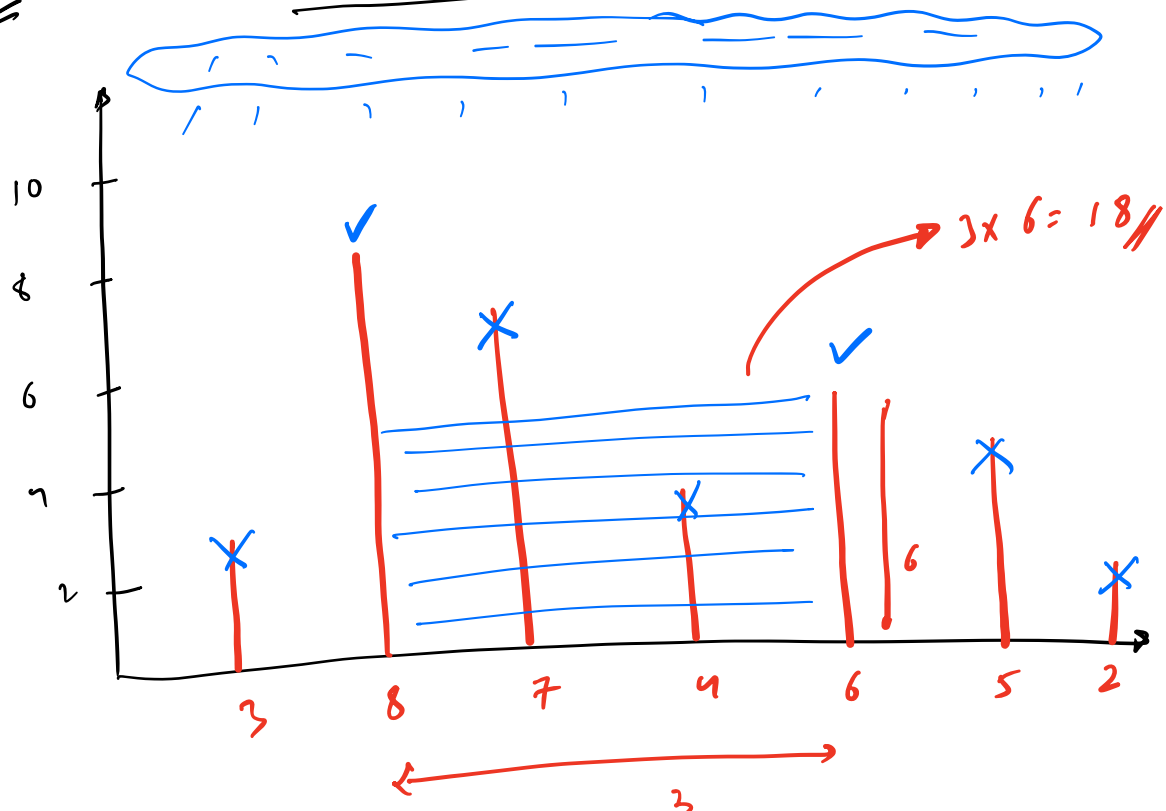
SC = O(1)

$$\begin{array}{r} +20 \\ \hline 39 \\ -1 \\ \hline 38 \end{array} \quad > 38 \quad == 38$$

---

Q

## Rain Water Trapping




find the pair of buildings that can contain the max amount of water [ every other building is removed ]

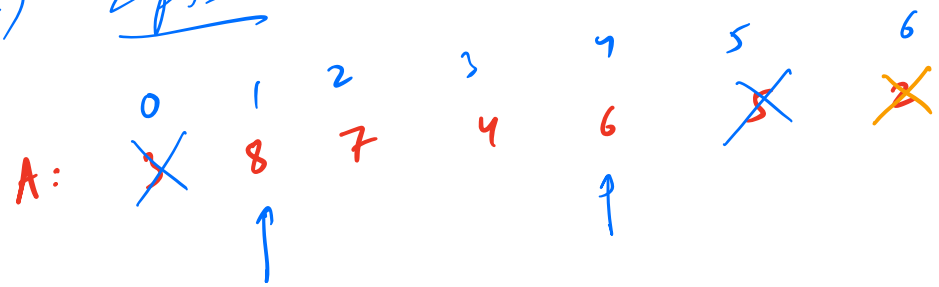
$(i, j) \longrightarrow$ width $= (j - i)$

height $= \min(A_i, A_j)$

$$\text{Water}(i, j) = (j - i) \times \min(A_i, A_j)$$

1) BF ⟶ $N^2$

2) 2 ptr




$(5-0)\times 3$

$5\times 3 = 15$

$(6-0)\times \min(3,2)$

$6\times 2 = 12$

ANS

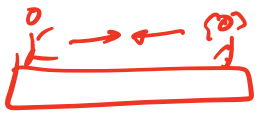
0 ~~4~~ ~~15~~ 20

$(5-1)\times 5$

$4\times 5 = 20$

$(4-1)\times(6)$

$3\times 6 = 18$



TC = O(N)

SC = O(1)

```
i = 0, j = N-1,   ANS = 0;
while( i < j ) {
    ANS = MAX(ANS, (j-i) x min(Ai, Aj));
    if (Ai < Aj) i++;
    else j--;
}
ret ans;
```