Q.1 Given a sorted Array of size N of distinct elements. Check if there exists a pair (i, j) s.t A[i] + A[j] = K , i != j.

↳ 2 Sum

A : {3, 7, 8, 11, 15} K = 14
↳ True
K = 20
↳ false

① Brute force : TC : O(N²)
SC : O(1)

② HashSet / HashMap ⇒ TC : O(N)
SC : O(N)

③ A : {3, 7, 8, 11, 15} K = 15
↑
i

Search K - A[i] ⇒ log N

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

A: {-3, 0, 1, 3, 6, 8, 11, 14, 18, 25} K=17

(0 1 2 3 4 5 6 7 8 9)

P_1 → 3, P_2 → 14

P_1	P_2	Sum	
0	9	22 > 17	⇒ P_2--
0	8	15 < 17	⇒ P_1++
1	8	18 > 17	⇒ P_2--
1	7	14 < 17	⇒ P_1++
2	7	15 < 17	⇒ P_1++
3	7	17 == 17	⇒ True.

#

A: {3, 7, 8, 11, 15} K = 20

(0 1 2 3 4)

P_1, P_2 → 11

P_1	P_2	Sum	
0	4	18 < 20	⇒ P_1++
1	4	22 > 20	⇒ P_2--
1	3	18 < 20	⇒ P_1++
2	3	19 < 20	⇒ P_1++

Code

```
{ P1 = 0, P2 = N-1
  while (P1 < P2) {
      ==
}
```

Q.2 Amazon/Visa Given a sorted Array of size N of distinct elements. Check if there exists a pair (i, j) s.t $A[i] - A[j] = K$, $K > 0$

	0	1	2	3	4	5	6	7	8	9
A:	-3	0	1	3	6	8	11	14	18	25
					j		i			

K=5
↳ true.

	0	1	2	3	4	5	6	7	8	9	
A:	-3	0	1	3	6	8	11	14	18	25	K=5
						P_1	P_2				

	P_1	P_2	$A[P_2] - A[P_1]$	
1)	0	N-1	28 > 5 ✗	P_1++ diff ↓ / P_2-- diff ↓
2)	0	$\frac{N}{2}$ (5)	11 > 5	P_1++ / P_2--
3)	N/2 (5)	N/2 + 1 (6)	3 < 5 ✗	P_2++ / P_1--
4)	0	1	3 < 5 → P_2++	

A: { -3, 0, 1, 3, 6, 8, 11, 14, 18, 25 } K = 5

(0 1 2 3 4 5 6 7 8 9)

P_1 = 1, P_2 = 3

P_1	P_2	$A[P_2] - A[P_1]$
0	1	$3 < 5 \uparrow \Rightarrow P_2{+}{+}$
0	2	$4 < 5 \uparrow \Rightarrow P_2{+}{+}$
0	3	$6 > 5 \downarrow \Rightarrow P_1{+}{+}$
1	3	$3 < 5 \uparrow \Rightarrow P_2{+}{+}$
1	4	$6 > 5 \downarrow \Rightarrow P_1{+}{+}$
2	4	$5 == 5 \Rightarrow$ ✓ True.

A: { 1 4 6 } K = 2

(0 1 2)

P_1 = 0, P_2 = 1

$3 > 2$ ↓

A: { 0 3 5 30 35 36 } K = 1

(0 1 2 3 4 5)

P_1 = 4, P_2 = 5

↳ true.

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

```
P1 = 0, P2 = 1
while ( P2 < N ) {
    diff = A[P2] - A[P1]
    if (diff == K) return true;
    if (diff > K) {
        P1++
        if(P1 == P2) P2++;
    }
    else  P2++
}
return false;
```

Step to solve 2 pointers :-

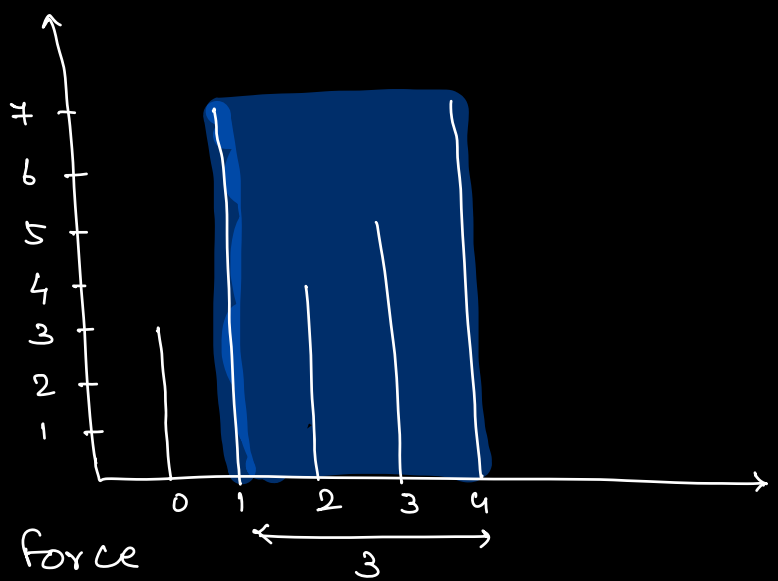
1. How to initialize the pointers.
2. How to move the pointers.
3. Stopping condition.

Q.3 Arcesium Swiggy Intuit

Rain Water Trapping

Given an Array of size N, where A[i] represents the height of ith wall. Pick any 2 walls s.t max water can be stored b/w them.

Ex: {3, 7, 4, 5, 7} (indices 0, 1, 2, 3, 4)

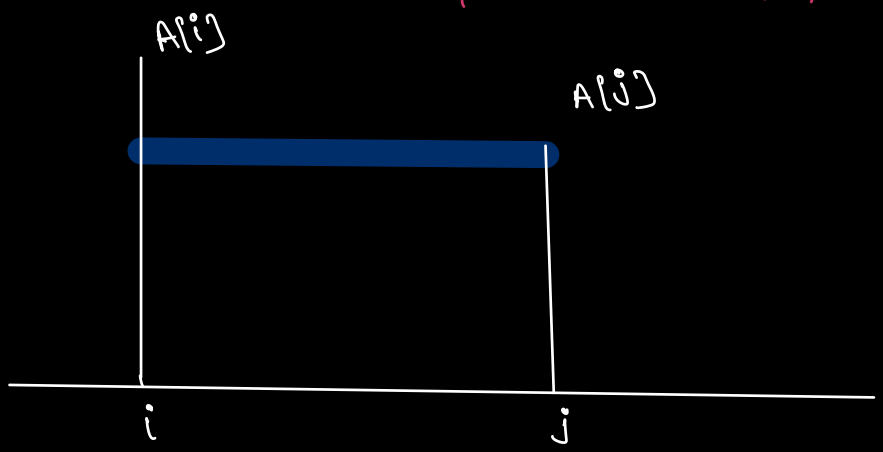


⇒ 21

Brute force

↳ Check for every pair of walls & find Max

⇒ TC: $O(N^2)$, SC: $O(1)$

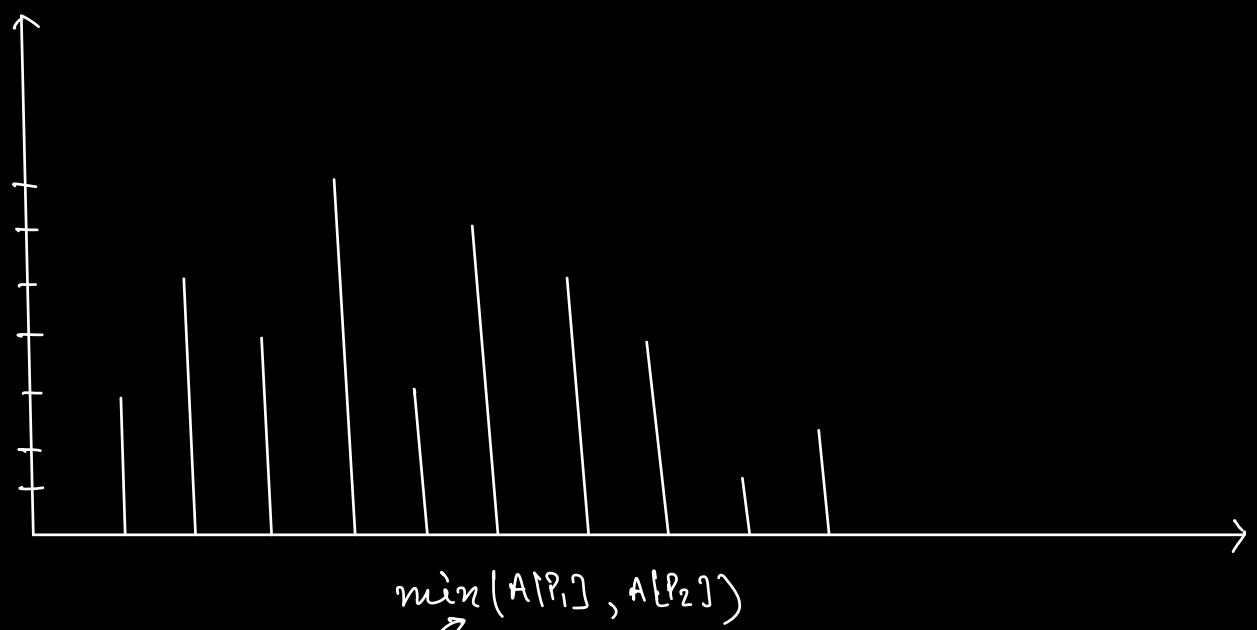


water = Ht * wd

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

	0	1	2	3	4	5	6	7	8	9
A:	3	5	4	4	3	6	5	4	1	2

p_1 p_2



p_1	p_2	Ht	Width	water
0	9	2	9	18
0	8	1	8	8
0	7	3	7	21
1	7	4	6	24
1	6	5	5	25
2	6	4	4	16
3	6	5	3	15

3	5	6	2	12
3	4	3	1	3

3 3 ⇒ 25 ans.

break.

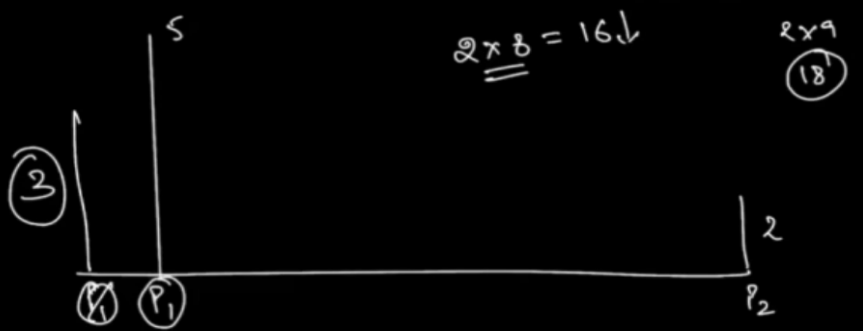
⇒ Move the pointer with lesser height.

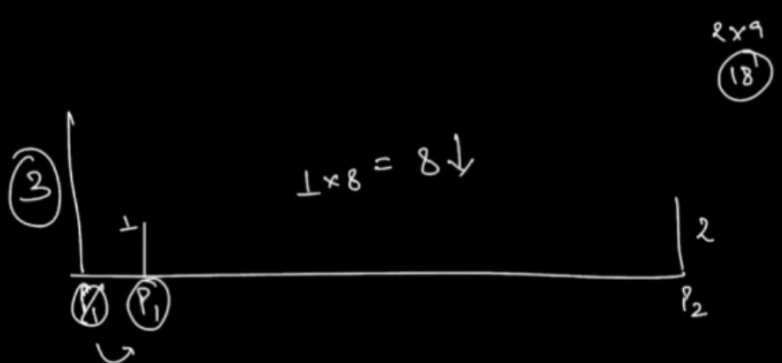
O(N)

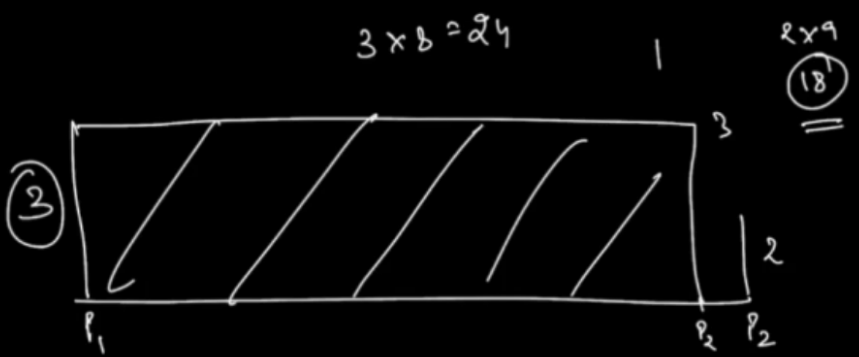
$P_1 = 0$

$P_2 = N-1$

Code: HW







Q.4 Given 3 sorted Arrays A[], B[] & C[] of sizes N. Find i, j & k s.t

MS.

$\max(A[i], B[j], C[k]) - \min(A[i], B[j], C[k])$ is minimized.

A: { 3 (0), 14 (1), 16 (2), 20 (3), 29 (4), 40 (5) }

B: { -6, 23, 24, 30, 35, 50 }

C: { -15, 15, 26, 31, 39, 42 }

Minimise ↓ diff

i	j	k	min	max	diff
0	0	0	-15	3	18
3	4	5	20	42	22
4	3	3	29	31	2

Brute force

```
for(i → 0 to N) ⇒ A
    for(j → 0 to N) ⇒ B
        for(k → 0 to N) ⇒ C
```

TC: $O(N^3)$

SC: $O(1)$

$X - Y \rightarrow$ maximise this

$\max(A_i, B_j, C_k)$ — $\min(A_i, B_j, C_k)$

$\Rightarrow$

A : { 3 (0), 14 (1), 16 (2), 20 (3) P_1, 29 (4), 40 (5) }

B : { -6, 23 P_2, 24, 30, 35, 50 }

C : { -15, 15, 26, 31, 39, 42 }
~~P_3~~ ~~P_3~~ P_3

P_1	P_2	P_3	max	min	diff → minimise this.
0	0	0	3	-15	18
0	0	1	15	-6	21
0	1	1	23	3	20
1	1	1	23	14	9
2	1	1	23	15	8
2	1	2	26	16	10
3	1	2	26	20	6
		⋮	1		
		1	⋮		

$p_1 < N \;\&\&\; p_2 < N \;\&\&\; p_3 < N$

⇒ Increment the pointer with min. Value.

$X - Y \rightarrow$ maximise this

$\max(A_i, B_j, C_k)$ ↑ X — Y ← $\min(A_i, B_j, C_k)$

⇒

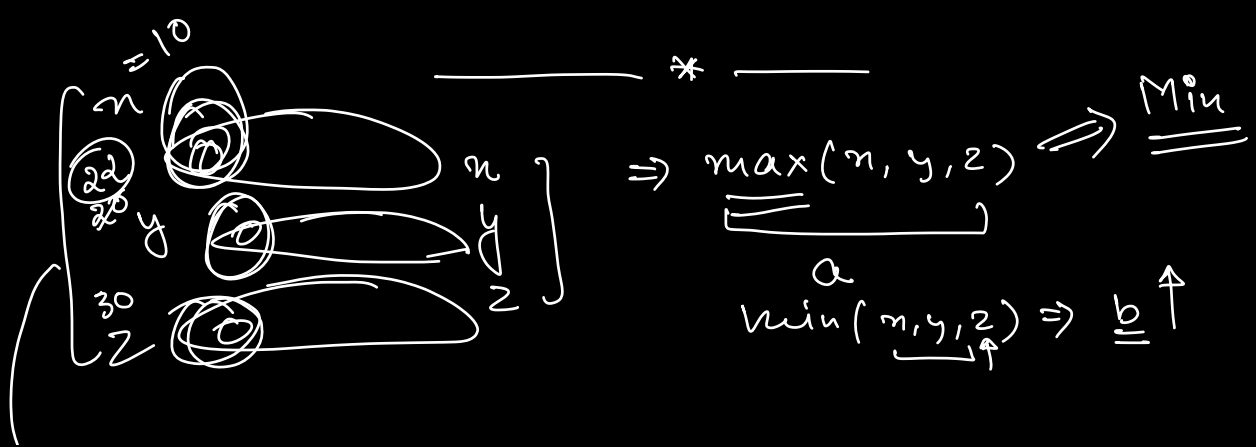
$p_1 = 0$
$p_2 = 0$
$p_3 = 0$

while($p_1 < N \;\&\&\; p_2 < N \;\&\&\; p_3 < N$) {

≡

TC: $O(N)$ ← $N+N+N$
SC: $O(1)$

}



min

min(x, y, z) ⇒ min

10 20 30

↳ 10 ⇒ 10