2 - Pointers

C++ address Pointers?

int *a, *b;

⇒ Physical Representation: Variables which hold index values

0	1	2	3	4	5
3	7	4	1	6	7

p1=2

2 Important Questions:

1) How (where) to initialize the pointers?

2) How to update the Pointers?

Question: 2 Sum

Given a sorted array, check if there is a pair (i,j) of distinct elements numbers such

$a[i] + a[j] == K$ s.t. $i \neq j$

$K = 17$

A =	-3	0	1	3	6	7	12	14	18	25
	0	1	2	3	4	5	6	7	8	9

$\boxed{(3, 14)}$ True

Brute Force:

consider all pairs.

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

T.C : $O(n^2)$
S.C : $O(1)$

True;

} false
}

Hashing :

T.C : $O(n)$
S.C : $O(n)$;

$\boxed{k = 24}$

a = 12

$\boxed{b = 12}$

k = 17

Binary Search :

-3	0	1	3	6	7	12	14	18	25
0	1	2	3	4	5	6	7	8	9

1) a + b = 17

a = -3 ⇒ $\boxed{b = 20}$ → Do B.S to check if 20 is there or not.

2) a = 0
b = 17 → O(log n)

T.C : O(n log n)
S.C : O(1)

Edge Case :
P_1 : 0 → P {P}
P_2 : A → P {n-P}

P + n - P = $\boxed{n}$

$P_1 = 3$, $P_2 = 7$

2 - Pointers :

$\boxed{K = 17}$

A =	~~-3~~	0	1	3	6	7	12	14	18	~~25~~
	0	1	2	3	4	5	6	7	8	9

P1 = 0, P2 = 9

$\boxed{\text{Count} = 1}$

(3, 14) = 17

(-3, 25) = 22

count++;

a[P1] + a[P2] = 22

> 22

n = 3 1

(,) = 15

$P_2 = 14$

(−3, 10)

< 15

(0, 18) = 18 > 17

(0, 14)

(3, 14)

(1, 14)

```
bool checkSum(arr[], n, k){
    p1 = 0, p2 = n-1;
    count = 1;
    while( p1 < p2 ){
        if( a[p1] + a[p2] < k )
            p1++;
        else if ( a[p1] + a[p2] > k )
            p2--;
        else {
            count++;
            p1++;  p2--;
        }
    }
    return false;  (return count;
}
```

$P_1 = 3$

$P_2 = 7$

k = 7

T.C:

Question: Count no of pairs

T.C: O(n)   S.C: O(1)

Question: Duplicate

K = 10

A:	1	4	4	4	5	5	5	6	6	11
	0	1	2	3	4	5	6	7	8	9

(4, 5)
(4, 6)
(5, 6)
} 3 pairs

(4, 6) => 6 pairs

(4+6) = 10

6 + 3 = $\boxed{9}$

4s = 3
6s = 2
} 3 × 2 pairs = 6 pairs

5s = 3

(5+5) = 10

3 × 3 = 9 ?

5 5 5 5

$\binom{^4C_2}{} =$

$^nC_2 = \frac{n(n-1)}{2}$

p1 = 0   p2 = n-1

#4s = 3 , #6s = 2

ans += 3 × 2 = $\boxed{6}$

k = 10

A = 1 4 4 4 5 5 5 6 6 11

(p2 - p1 + 1)

No of 5's

count = $^xC_2$

(1, 11) > 10

(1, 6) < 10

Case 1: (4, 6) = 10

Case 2: (5, 5) = (Do we have to greater)

p1 . . . p2

6 6 6 6 6 6

```
if ( a[p1] + a[p2] == k ) {
    if ( a[p1] != a[p2] ) {
        count_p1 = 0, left = p1;
        while ( a[p1] == a[left] ) {
            p1++;
            count_p1++;
        }
        count_p2 = 0; right = p2;
        while ( a[p2] == a[right] ) {
            p2--;
            count_p2++;
        }
        count += count_p1 * count_p2;
    }
    else ( a[p1] == a[p2] ) {
        count += (p2 - p1 + 1)C2;
        return count;
    }
    return count;
}
```

$^xC_2$ $\left(\frac{x(n-1)}{2}\right)$

T.C: O(n)

S.C: O(1)

Question: Given sorted array of distinct elements, find a pair (i, j) such that (a[j] - a[i] = K) (i < j)

K = 11

A =	-3	0	1	3	6	7	12	14	18	25
	0	1	2	3	4	5	6	7	8	9

(14, 3) = 11

→ If there is a pair with difference = K, Is there a pair whose diff = -K

(a, b) ⇒ a - b = $\boxed{\boxed{K}}$ ← $\boxed{x}$

(b, a) ⇒ b - a = -K ← x

$\boxed{abs(K)}$

$\boxed{-5}$ ⇒ $\boxed{5}$

Brate Force:

T.C: $O(n^2)$
S.C: $O(1)$

Binary Search:

T.C: $O(n \log n)$
S.C: $O(1)$

Hashing:

T.C: $O(n)$
S.C: $O(n)$

2 - Pointers:

How to Initialize?

How

1) 0, n−1
2) $\frac{n}{2}$, $\frac{n}{2}-1$
3) 0, 1 {n, n−1}
4) 0, $\frac{n}{2}$

K = 5

					P_2	P_1				
A =	−3	0	1	3	6	7	12	14	18	25
	0	1	2	3	4	5	6	7	8	9

→ p1++ } Brute force
→ p2++ }

1) $P_1 = 0$, $P_2 = 9$

25 − (−3) =

(25, −3) ⇒ 28 ⇒

p1++ ⇒ (0, 25) = 25

p2-- ⇒ (−3, 18) = 21

2) $P_1 = 5$, $P_2 = 4$

(7, 6) ⇒ 1

(7, 3) ⇒ 4

✓ p1--;

(12, 6) ⇒ 6

✓ p2++;

3) $P_1 = 0$, $P_2 = 1$

$\boxed{K = 5}$

			P_1		P_2					
A =	−3	0 P_1	1	3	6	7	12	14	18	25
	0	1	2	3	4	5	6	7	8	9

(−3, 1) = 4

(−3, 0) ⇒ 3

(−3, 1) ⇒ 4

(−3, 3) ⇒ 6

(0, 3) ⇒ 3

(0, 6) ⇒ 6

(1, 6) ⇒ 5

→ No. of pairs ✓ }
→ Duplicates ✓

Takeaways

1) Initialization of Pointers:
Initialize pointers such that you can always make a decision and discard an element

2) Update the Pointers
update towards your decision making

3) when to stop

Binary Search:

1) low = 0, high = n-1
2) low = mid+1 (p_1) or high = mid - 1 (p_2)
3) low > high

→ Increment p1 & decrement p2

→ $\boxed{\text{2 Pointers is superior}}$

Sum of 3 nos

Question: Given an array of integers, find three integers from the array such that 3 integers sum is closest to B.

the sum of these 3 numbers

→ Return sum of these 3 numbers

$\boxed{B = 14}$

A =	7	0	-3	16	-10	12	-7	3
	0	1	2	3	4	5	6	7

(7, 16, -10) ⇒ (13) (14-13) : 1

(-3, 0, 16) ⇒ 13 (14-13) : 1

(0, 3, 12) ⇒ 15 (14-15) : abs(-1) = 1

Simpler Problem :

Find 2 numbers whose sum is closest to B.

B = 14

A = 7 0 -5 16 -10 12 -7 3

(12, 3) (16, -3)

1 1

Brute Force :

Consider all pairs

T.C : $O(n^2)$

(0, 16) ⇒ 16, |-2|

(, 16) > 16

19 = |-5|

2 Pointer :

O(nlogn) $\boxed{B = 14}$

→ Sort the Array :

-10	-7	-3	0	3	7	12	16
0	1	2	3	4	5	6	7

P1 P1 P1 P2 P2

(-10, 16) ↓ 6 ⇒ $\boxed{8}$

(-10, a) 4 ⇒ diff

(-10, 16) = 6 (Diff = 8)

(-7, 16) = 9 Diff = 5

(-3, 16) = 13 Diff = 1

Diff = -2 (-5)

(-10, 16) = 6

$\boxed{<6}$ $\boxed{14}$ 10

(0, 16) = 16 (19)
(0, 12) = 12
(3, 12) = 15
(3, 7) = 10
(7, 7)

Diff =
Diff = 4
Diff = -1
Diff = 4
$P_1 = P_2$

(0, 16) = ...
sum = 18
-2
-4

min_diff = INT_MAX, sum;
p1 = 0, p2 = n-1;
while (p1 < p2) {
  if(abs(a[p1] + a[p2] - B) < min_diff) {
    min_diff = abs(a[p1] + a[p2] - B);
    sum = a[p1] + a[p2];
  }
  if(a[p1] + a[p2] < B)
    p1++;
  else p2--;
}

T.C: O(n) + O(nlogn)
S.C: O(1)

Let's Extend this to 3 numbers:

B = 14

A = -10 -7 -3 0 3 7 12 16

a, P_1, P_2

(16, -3, 0) (16, -10, 3)

minimum |a + b + c - B|

a + b + c = B

-10 + b + c = 14
$\boxed{b + c}$ = 24 (14)

Step1 : a = -10

(b + c - 10)

such that sum

⇒ Find 2 numbers b, c as close as possible to 14+10 = 24

Step2 : $a = -7$

$b + c = 14 + 7$
$= 21$

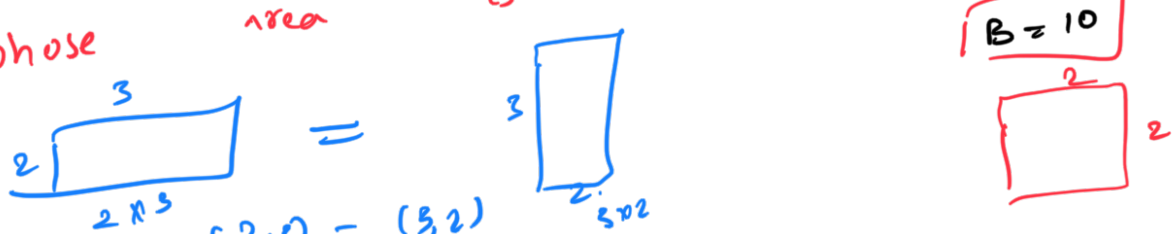
```
for (i=0; i<n; i++) {
      p1 = i+1;      p2 = n-1;
   O(n) ← [ < same code >
}
```

T.C : $O(n\log n) + O(n^2)$

$= \boxed{O(n^2)}$

Question: Sorted Array where elements represent lengths. Find no. of rectangles that can be formed using array as length and breadth whose area is less than B.

$\boxed{B = 10}$



$(2,3) = (3,2)$

A = { 2   3   5 }

(2, 2) =	4 < 10	Y	$\boxed{\text{Ans} = 3}$
(2, 3) =	6 < 10	Y	
		N	

$\boxed{(3,2) = (2,3)}$

(2,5) = 10
(3,3) = 9 < 10 Y
(3,5) = 15 N
(5,5) = 25 N

B = 85

	0	1	2	3	4	5	6	7
A =	2	3	7	8	10	12	19	20

2 => (2,2) (2,3) (2,7) (2,8) (2,10) (2,12) (2,19) (2,20) = 8
3 => (3,3) (3,7) (3,8) (3,10) (3,12) (3,19) (3,20) = 7
7 => (7,7) (7,8) (7,10) (7,12) = 4
8 => (8,8) (8,10) = 2
10 =>
12 =>

8 + 7 + 4 + 2 = $\boxed{21}$

Brute Force:

Consider all Pairs

T.C: $O(n^2)$

count += $(P_2 - P_1 + 1)$ = 4

$\boxed{(7,7)}$ (7,8) (7,10) (7,12)

(7,7) (8,7) (10,7) (12,7)

Approach 2:

B = 85

	P_1							
	2	3	7	8	10	12	19	20
	0	1	2	3	4	5	6	7

P_1, P_2

(2,20) = 40
(3,20) = 60
(7,20) = 140
(7,19) = 133
(7,12) = 84
96

count += P2 - P1 + 1
count = 8 + 7
= 15 + 4
= 19 + 2
= 21

$(P_1 > P_2)$ Break

(8, 12) =

(8, 10) = 80

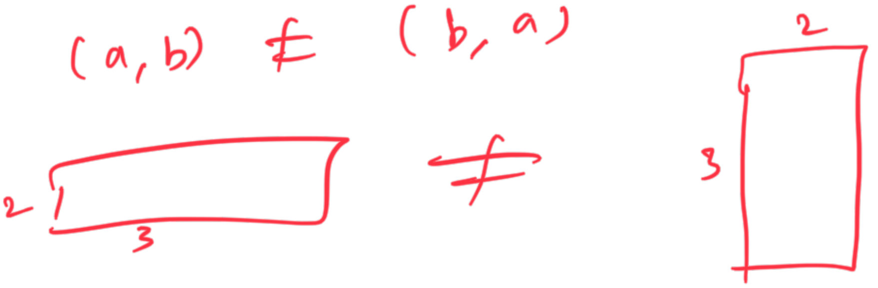
(10, 10) = 100

```
p1 = 0          p2 = n-1;
count = 0;
while (p1 < p2) {
    if (a[p1] * a[p2] < B) {
        count += p2 - p1 + 1;      [2(p2 - p1 + 1) - 1]
        p1++;
    }
    else
        p2--;
}
return count;
```

$\boxed{T.C: O(n)}$

$O(n \log n)$

$(a, b) \neq (b, a)$



Container with most Water

Question:

→ Given an unsorted array of elements which represent height of buildings, choose 2 buildings such that water trapped between the 2 buildings is maximum

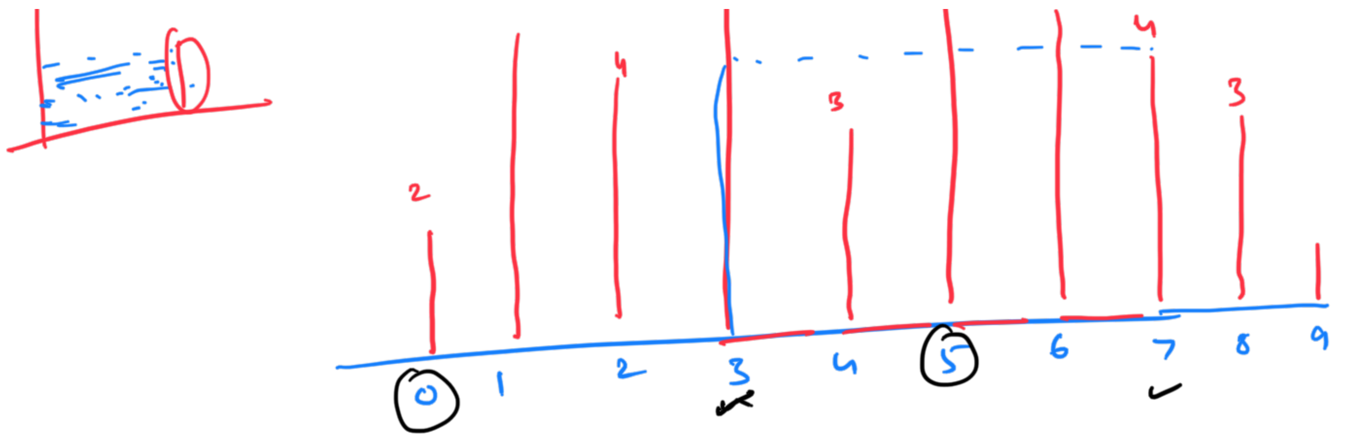
Delete / Remove $(n-2)$ buildings

$N = 10$

→

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

1   5   6   5



width = (5-0) = 5
h = min(2, 6) = 2
Area = 5 × 2 = 10

width = 4
height = min(7, 4)
Area = 4 × 4 = 16

Given 2 building (i, j)

width = j - i

height = min(a[i], a[j])

Area = (j-i) × min(a[i], a[j])

Brute Force:

Consider all pairs of buildings

```
for(i=0; i<n; i++){
    for(j=i+1; j<n; j++){
        ans = max(ans, )
    }
}
```

T.C: $O(n^2)$, S.C: $O(1)$

h = min(2, 1) = 1
min(9, 1) = 1

(2, 3) = 8 × 2 = 16

7 × 3 = 2

Better Approach:

w = 9
h = 1

P1 (index 3); P2 (indices 6, 7, 8, 9)

A =										
value	2	3	4	7	3	6	5	4	3	1
index	0	1	2	3	4	5	6	7	8	9

9 × 1 = $\boxed{9}$

Observation:

Area = (0, 9) => 9 × min(2, 1) = 1 × 9 = 9

h = 1   w = 9

I can't get an area > 9 with 1 as one of the buildings

h = 1
h ≤ 1 , (w = < 9)
area = < 9

min(1, 7) = 1
min(0, 1) = 1

$\boxed{max\_area = 21}$

```
area = INT_MIN;
P1 = 0,   P2 = n-1;
while (P1 < P2) {
    area = max(area, min(a[P2], a[P1]) * (P2 - P1));
    if(a[P1] < a[P2])
        P1++;
    else
        P2--
}
return area;
```

T.C: $\boxed{O(n)}$

Median of Matrix of Sorted Rows

→ Median of Matrix of Sorted Rows
→ Sliding Window Technique
→ B.S Prob

B = 22

A =	-10	-7	-3	0	3	7	12	16
	0	1	2	3	4	5	6	7

(16, -3, 0) → (16, -10, 3)

Hashset = {{-10, -7, -3, 0, 3, 7, 12, {16}}}
(-10 → 0, 1; 16 → 7, 0, 1)

16 - 10 = $\boxed{6}$ 17 - 6 = $\boxed{7}$

22 - 6 = $\boxed{16}$

S.C: O(N)

(16 - 10) = 6

$(-10, 16, 16)$

T.C: $O(n)$

S.C: $O(n)$

$A =$

3	3	1	5	4
0	1	2	3	4
P_1	P_1			P_2

max_area = 12

Step1: Area: $4 \times \min(3,4) = 12$

Step2: Area: $3 \times \min(3,4) = \boxed{9}$