1) Pair with given Sum
2) Pair with given difference
3) Subarray with given sum
4) Container w/t most water

Q Given a sorted integer array A of size N.
Find any pair (i, j) s.t. $A[i] + A[j] = \boxed{K} \Rightarrow$ Given
$(i \neq j)$

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

$K = 11 \Rightarrow (2, 4)$

Sol^n 1) Brute Force

⇒ ∀ pairs (i, j) check if $(A[i] + A[j]) == K$.

```
for (i = 0 to N-1) {
    for (j = i+1 to N-1) {
        if (A[i] + A[j] == K) {
                return (i, j);
        }
    }
}
```

Use Binary Search to find (K-A[i]) since A is sorted

$T.C. = O(N^2)$

2) Binary Search

```
for (i=0; i<N; i++) {          N
        s = i+1;
        e = N-1;                 } log N
        target = K- A[i];
        ( B.s. )
}
```

$T.C. = O(N \log N)$

$S.C. = O(1)$

3) Two Pointer

↳ 2 variables pointing to 2 indexes of the array.

Q1 Where to initialise the 2 pointers ??

Q2 How to move the 2 pointers.

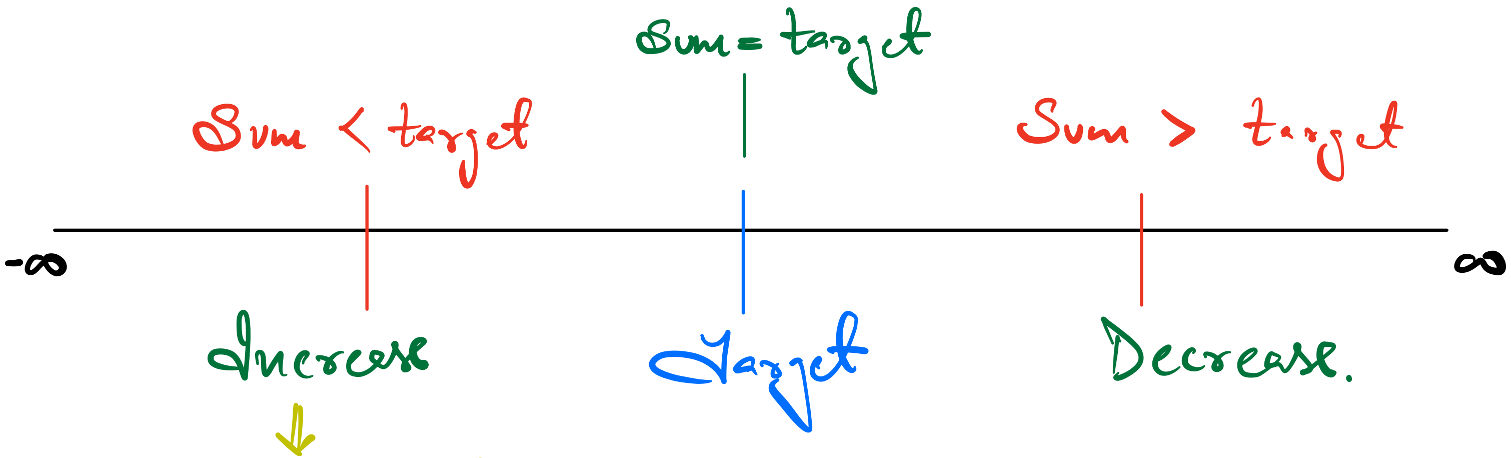
$$\begin{array}{cccccccc} & 0 & 1 & 2 & 3 & 4 & \cancel{5} & \cancel{6} \\ A = [& \cancel{-5}, & \cancel{-2}, & 1, & 8, & 10, & \cancel{12}, & \cancel{15}] \\ & \cancel{i} & \cancel{i} & i & & j & \cancel{j} & \cancel{j} \end{array}$$

$K = 11$

i	j	Sum
0	6 (N-1)	10 < 11 (Discard i)
1	6	13 > 11 (Discard j)
1	5	10 < 11 (Discard i)
2	5	13 > 11 (Discard j
2	4	(11 = 11)

Ans

Sum = A[i] + A[j]



i↑ → A[i]↑

⇓

Sum↑

j↓ ⇒ A[j]↓

⇓

Sum↓

Code

```
i=0,  j= N-1
while  (i<j) {
      Sum  =  A[i]+A[j];
```

```
        if (sum == K) {
                return (i, j);
        }
        else if (sum > target) {
                j--;
        } else {
                i++;
        }
}
return <-1, -1>
```

T.C. = O(N)

S.C. = $\underline{O(1)}$

Given a sorted integer array A of size N. find count of pair $\underline{(i, j)}$ s.t. $A[i] + A[j] = \boxed{K} \Rightarrow$ Given

$(i \,!= j)$

$$A = \begin{array}{c} \begin{array}{ccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left[\begin{array}{ccccccc} 1, & 2, & 3, & 4, & 5, & 6, & 8 \end{array}\right] \end{array}$$

$\underline{K = 10}$

Ans = 2 ((1,6), (3,5))

Case-1 Distinct Elements

Code

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

T.C. = O(N)

S.C. = O(1)

Case-II Duplicates

K= 10

freq i   freq j

$$A = \left[\begin{array}{cccccccccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 1, & 1, & 1, & 2, & 3, & 3, & 3, & 3, & 4, & 5, & 5, & 5, & 5, & 7, & 7, & 7 \\ \cancel{i} & \cancel{i} & \cancel{i} & \cancel{i} & \cancel{i} & & & & \cancel{i} & i & & & j & & & \cancel{j} \end{array}\right]$$

A[4] + A[13] = 10(K)

(4, 11)	(5,11)	(6, 11)	(7, 11)	
(4,12)	(5,12)	(6,12)	(7,12)	12
(4,13)	(5,13)	(6, 13)	(7,13)	

$$^4C_2 = \frac{\cancel{4}\times 3}{\cancel{2}} = \boxed{6}$$

$+$

18

Q Given a sorted array of size N & an integer $(K>0)$
find any pair (i, j) s.t. $(A[j] - A[i]) == K$
$(i \,!\!= j)$

$$A = \left[\begin{array}{ccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ -5, & -2, & 1, & 8, & 10, & 12, & 15 \end{array}\right]$$

$K = 11 \Rightarrow \underline{(\overset{i}{2}, \overset{j}{5})} \Rightarrow (12 - 1) = 11$

Ans

1) Brute force → Check all pairs.

$T.C. = O(N^2)$

$(A[j] - A[i]) > \underline{0} \rightarrow A[j] > A[i]$

$\Downarrow$

$(j \geq i)$

$\underline{K>0}$

2) Binary Search

```
for (i=0; i<N; i++) {            N
        s = i+1;                 }
        e = N-1;                 }  log N
        target = K + A[i];       }
        ( B.S. )
}
```

T.C. = $O(N \log N)$

S.C. = $O(1)$

3) Two Pointers

Q1 Where to initialise two pointers

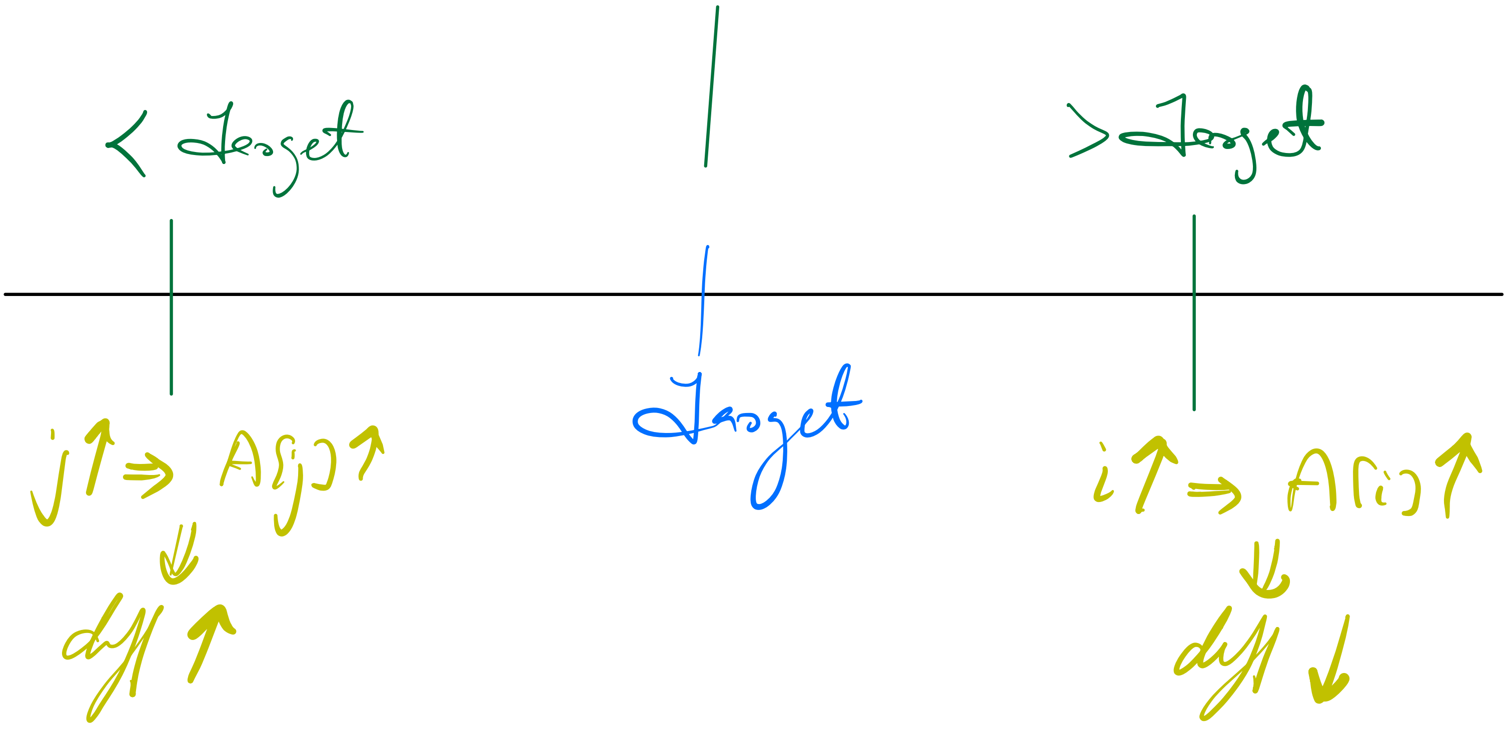
1) $i=0$, $j=N-1$ ✗ ⇒ Sum
2) $i=mid$, $j=mid+1$ ✗
3) $i=0$, $j=1$ ✓ ⇒ Diff.

Q2 How to move the pointers.

$$A = \left[\begin{array}{ccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ -5, & -2, & 1, & 8, & 10, & 12, & 15 \\ & & & i & j & & \end{array}\right]$$

$K = 11$

i	j	diff		
0	6 (N-1)	20 > 11		×
0	5	17	j--	diff ↓
1	6	17	i++	diff ↓
3 (mid)	4 (mid+1)	2 < target		×
2	4	9	i--	diff ↑
3	5	4	j++	diff ↑



$$A = \left[\begin{array}{ccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ -5, & -2, & 1, & 8, & 10, & 12, & 15 \\ i & & & & & & \end{array}\right]$$

$K = 11$ $\quad\quad\quad\quad j$ $\quad\quad$ $diff = A[j] - A[i]$

i	j	diff	
0	1	$-2-(-5) = 3 < 11$	⇒ j++
0	2	$1-(-5) = 6 < 11$	⇒ j++
0	3	$8-(-5) = 13 > 11$	⇒ i++
1	3	$8-(-2) = 10 < 11$	⇒ j++
1	4	$10-(-2) = 12 > 11$	⇒ i++
2	4	$10-1 = 9 < 11$	⇒ j++
2	5	$12-1 = \boxed{11 = 11}$	

Code

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

T.C. = O(N)
S.C. = O(1)

Q: Given an integer array of +ve elements Check if a subarray with sum = K is present.

$$A = \begin{array}{ccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 1, & 3, & 15, & 10, & 20, & 3, & 23 \end{array}$$

K = 33 ⇒ True

1) PS + Hashing

2) Optimal

$K = 33 \Rightarrow$ True

$$A = \begin{array}{ccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1, & 3, & 15, & 10, & 20, & 3, & 23 \end{array}$$

$PS: = [1, 4, 19, 29, 49, 52, 75] \Rightarrow$ Sorted

$$\text{Sum}[i, j] == K$$

$\Downarrow$

$PS[j] - PS[i-1] == K$ (if $i != 0$)

$(PS[j] == K)$ (if $i == 0$) loop

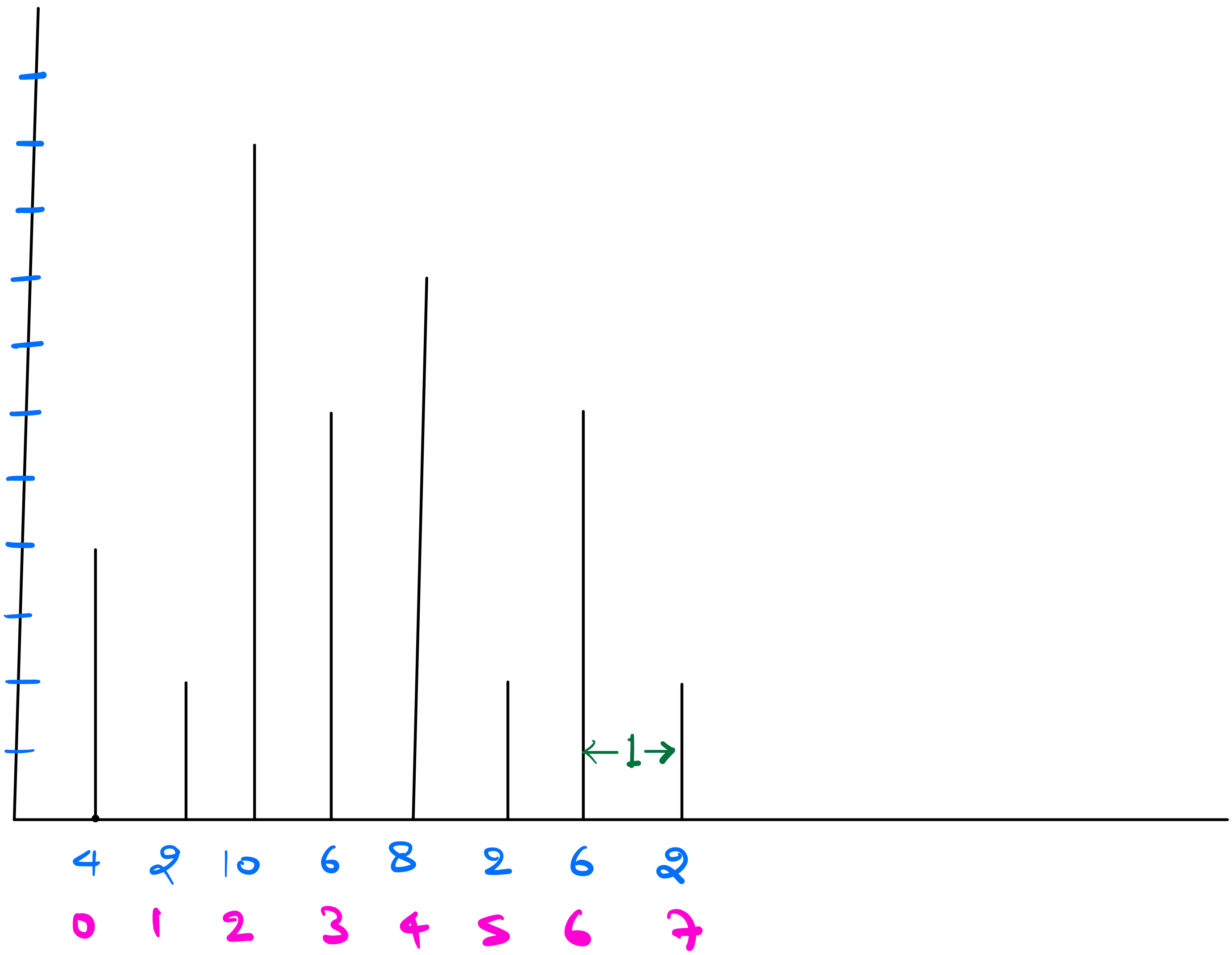
Q// Given an integer array of size N. representing the height of N walls.

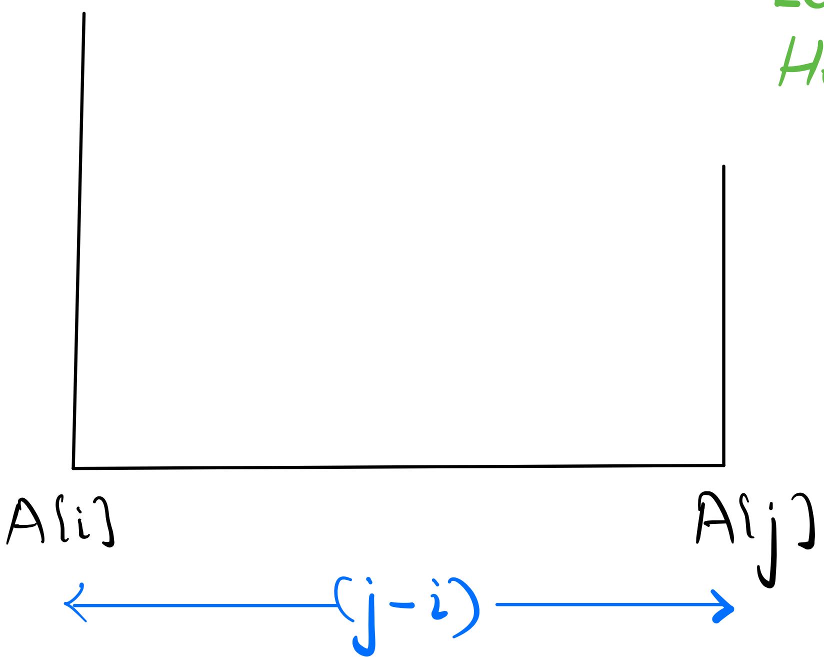
Find any 2 walls that can form a container to store max amount of water

1) Volume = Area ←

2) The distance b/w every consecutive wall is 1 unit

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





Length = (j-i)
Height = min(A[i], A[j])

$$\text{Amount of water} = (j-i) \times \min(A[i], A[j])$$

1) Brute force

∀ i, j where $i<j$ ⇒ Calculate the area & take max.

$$T.C. = O(N^2)$$
$$S.C. = \underline{\underline{O(1)}}$$

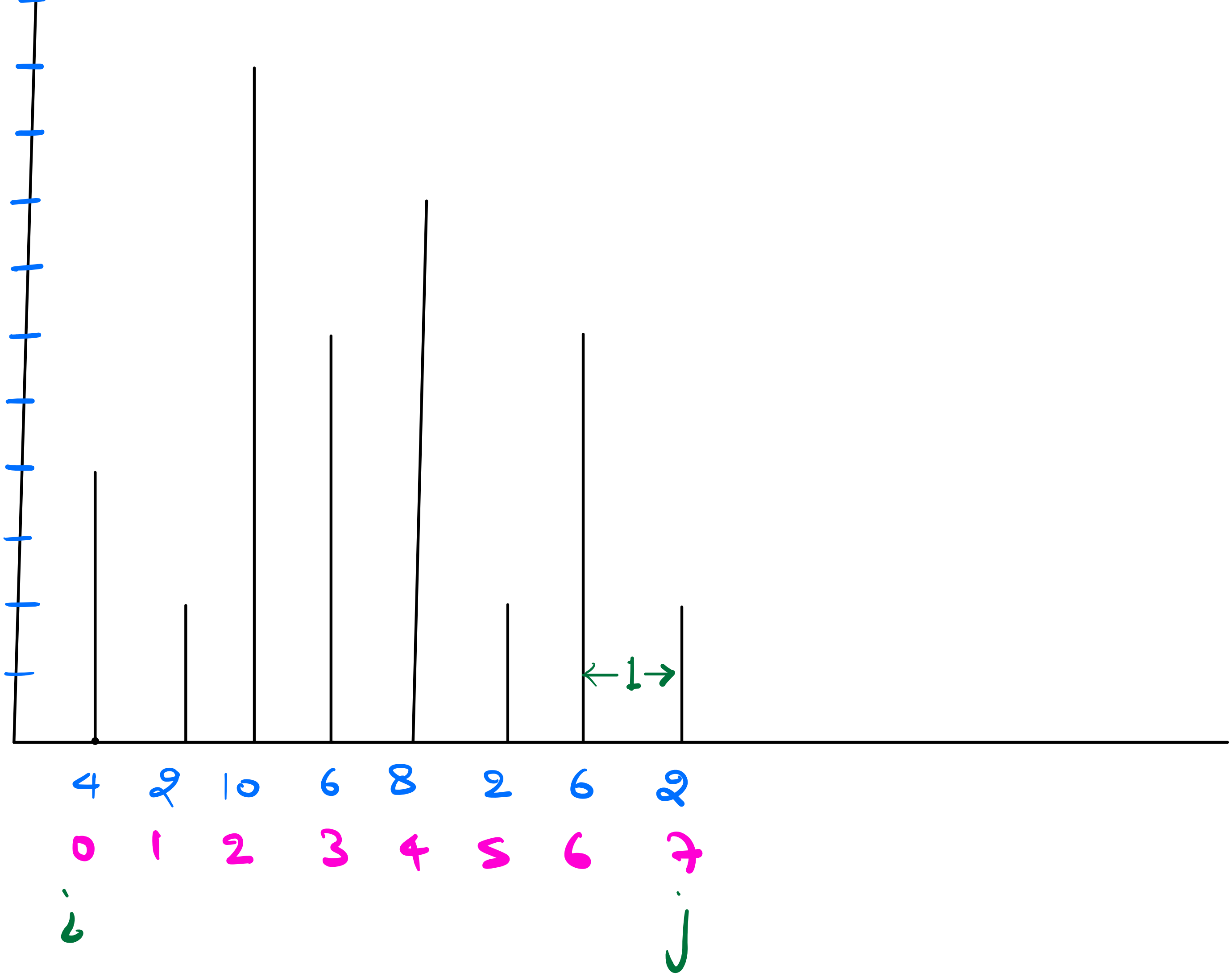
2) Optimal Solution

$$\text{Amount of water} = \overset{\text{Length}}{\underline{(j-i)}} \times \overset{\text{Height}}{\underline{\min(A[i], A[j])}}$$

Goal : Maximise amount of water.

Since we need to maximize area. We start with max length.

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



i	j	Length	Height	Amount
0	7	7	min(4,2) = 2(j)	14

⇓ ⇓

move / discard smaller wall

Code

```
i=0,   j=n-1;
ans = INT_MIN
while (i<j) {
        area =  min (A[i], A[j]) × (j-i)
        ans =  max (area, ans);

         if (A[i] < A[j]) {
                   i++;
         } else if (A[j] < A[i]) {
                   j++;
         }
         else {
                   i++;
                   j++;
         }
}
return ans;
```