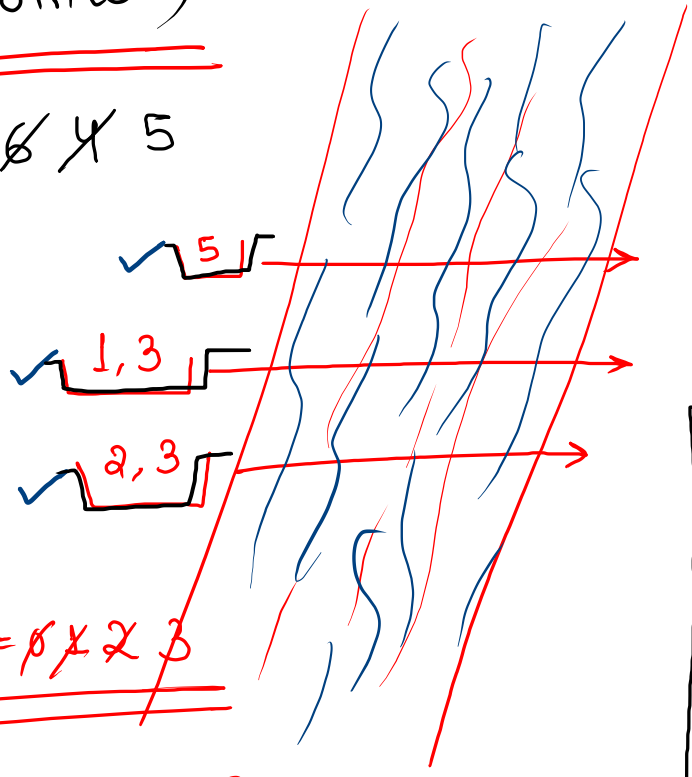
Count boat

limit = 5

arr :- [3, 5, 1, 2, 3]

(2 pointers)

sum = ~~6~~ ~~4~~ 5



Count = ~~0~~ ~~1~~ 2 3

ans = 3

0 1 2 3 4

arr :- [1, 2, 3, 3, 5]

j i

```
Sort(arr);
while ( i <=j ) {
    int sum = arr[i] + arr[j];
    if ( sum <= limit){
        i++, j--;
    }else{
        j--;
    }
    Count++;
}
```

HW_Plant Watering by Alice and Bob

arr :- [2 , 2 , 3 , 3]

b a

cA = ~~5~~ ~~3~~ 1

cB = ~~5~~ ~~2~~ ~~5~~ 2

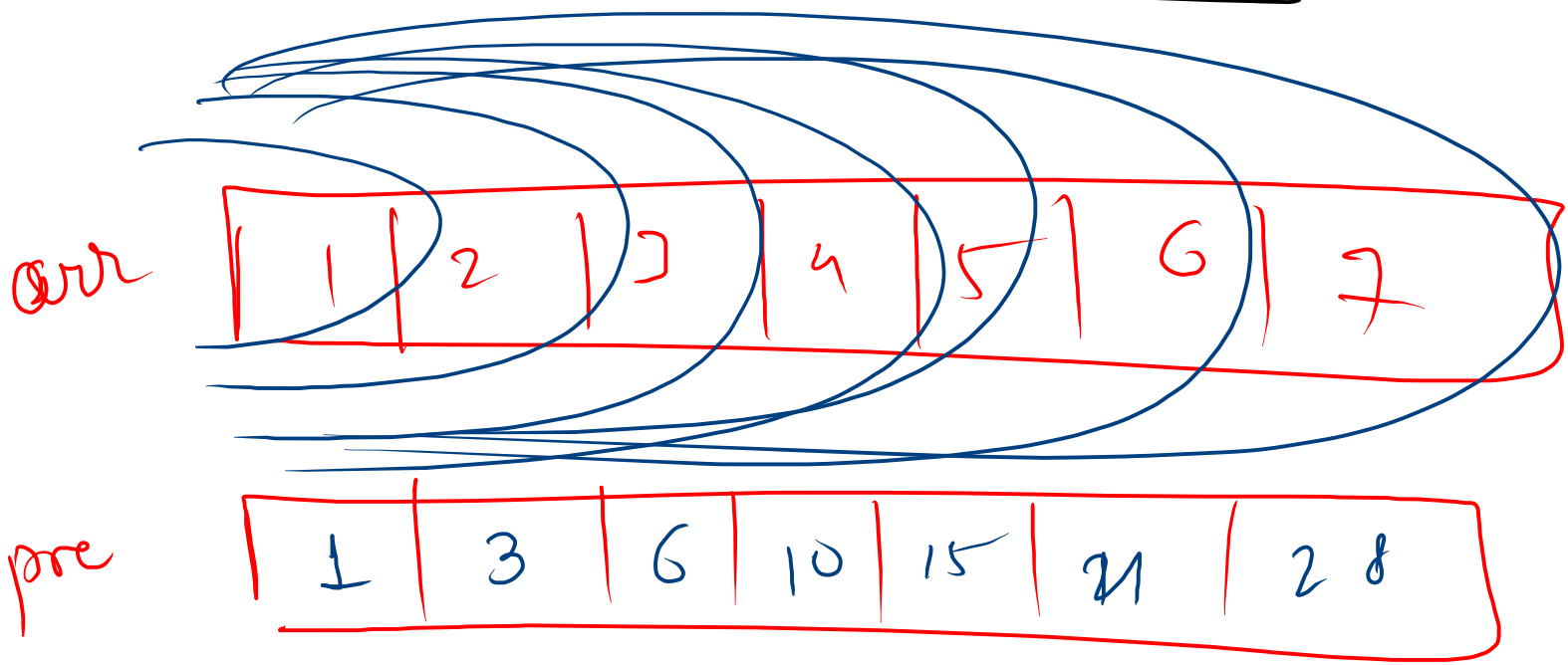
Count = ~~0~~ 1

Revision :-

- ✓ sorting, lambda function
- ✓ arrays, subarrays (Kadan's algo)
- ✓ 2 pointers
- ✓ Prefix array
- ✓ Arrays as Hashmap
- ✓ 2d arrays
- ✓ Strings, substring
- Binary Search (BSLB, BSUB)
- Arraylist
- Stack
- Hashmap, hashset
- Queue
- PQ

→ Prefix sum

sum of all ele. on left including itself



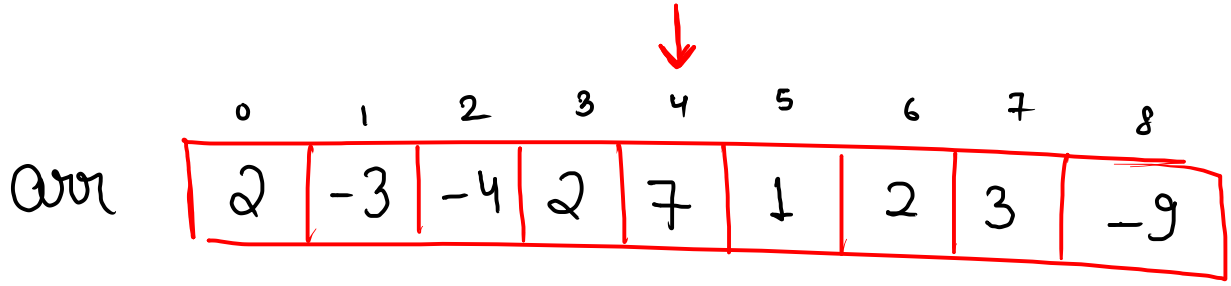
prefix sum

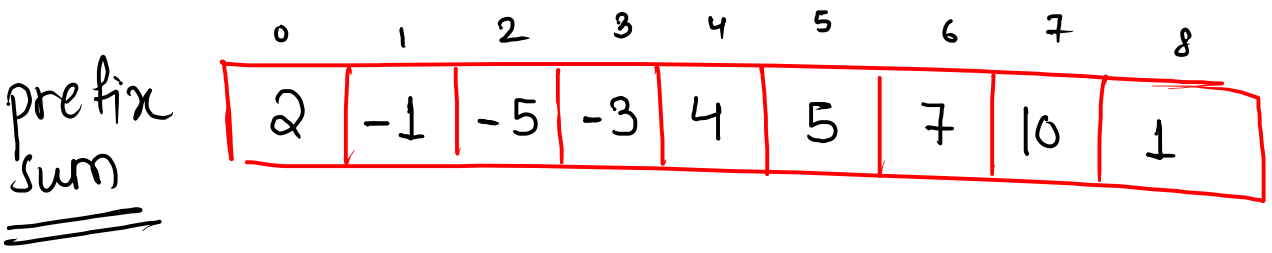
$pre[i] = pre[i-1] + arr[i]$

suffix sum

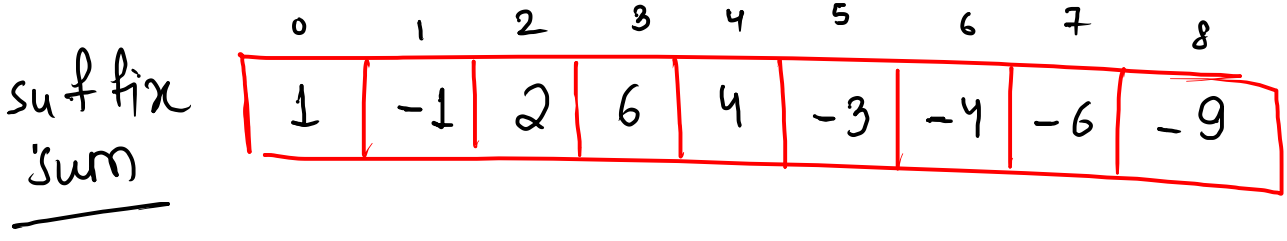
$suffix[i] = suffix[i+1] + arr[i]$

Find Pivot Index 1





ans = 1



⇒ array as hashmap

HW_Lucky Number 26

{ bubble
insertion
selection

arr =

1	2	3	4	3	4	2	4	4	6	5

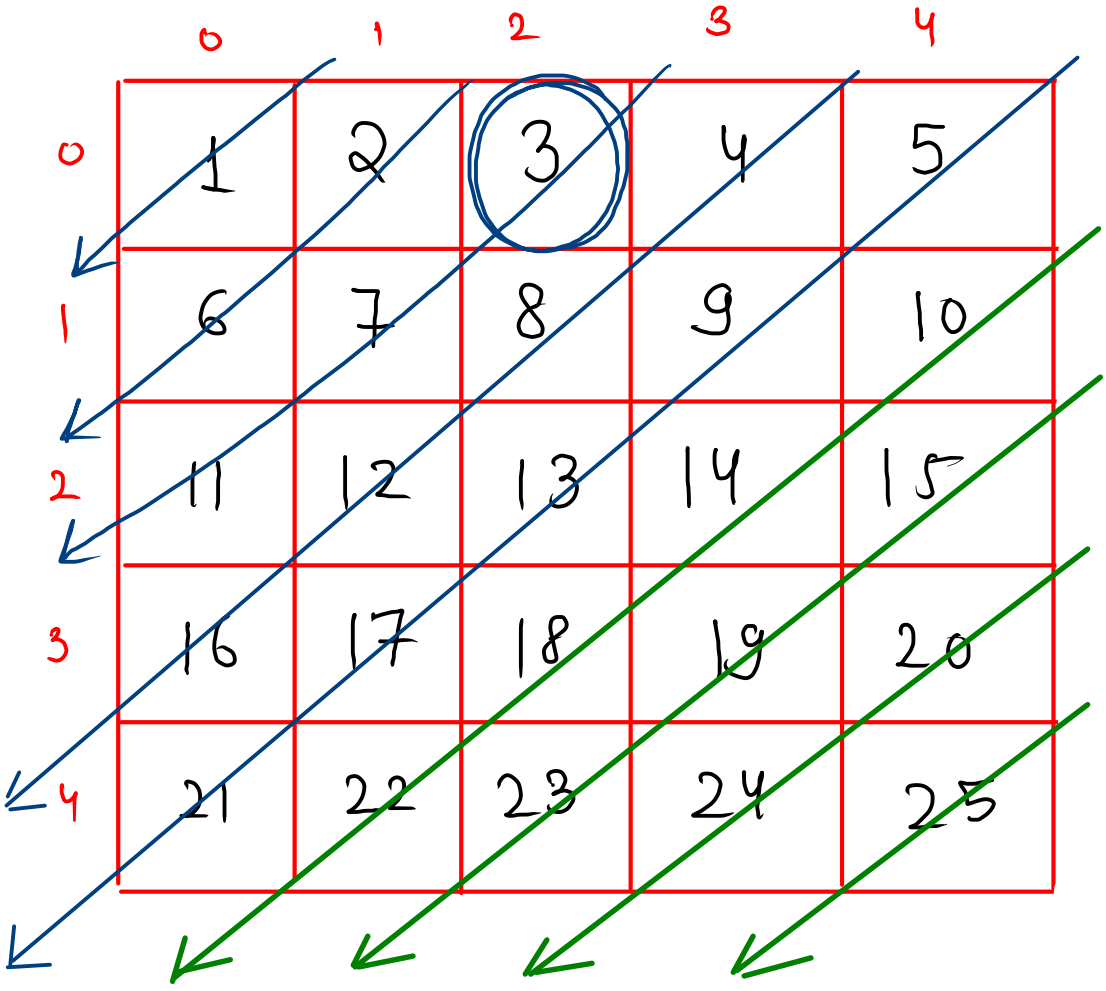
{ Quick
Merge

freq =

0	1	2	3	4	5	6	7	8	9
0	~~0~~ 1	~~0~~ ~~1~~ 2	~~0~~ ~~1~~ 2	~~0~~ ~~1~~ ~~2~~ ~~3~~ 4	~~0~~ 1	~~0~~ 1	0	0	0

{ count
radix
bucket

Print the matrix left-diagonal wise



i j

(0, 2)

(1, 1)

i++ j

j--

j >= 0

i<n

(0,0)	(1,4)
(0,1)	(2,4)
(0,2)	(3,4)
(0,3)	(4,4)
(0,4)	

i = 1 → 4, j = 4

```
for(int gap=0; gap<n; gap++){
    for(int i=0, j=gap; j>=0; i++, j--){
        Syso(arr[i][j])
    }
}
```

```
for(int gap=1; gap<n; gap++){
    for(int i=gap, j=n-1; i<n; i++, j--){
        Syso(arr[i][j])
    }
}
```

Transpose of Matrix of N*N

	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15
3	16	17	18	19	20
4	21	22	23	24	25

$(i, j) \leftrightarrow (j, i)$

```
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        if ( i > j ) {
            int temp = arr[i][j];
            arr[i][j] = arr[j][i];
            arr[j][i] = temp;
        }
    }
}
```

⇒ StringBuilder

Array ⟶ ArrayList

String ⟶ StringBuilder

Syntax

```
StringBuilder sb = new StringBuilder();  // mutable
```

Inbuilt fn

- sb.append(ch);
- sb.deleteCharAt(idx);
- sb.reverse();
- sb.charAt(idx);

Convert 1-D Array to 2-D Array

arr1d = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]

idx: 0 1 2 3 4 5 6 7 8 9 10 11

p = 2

q = 6

$$i = idx / q$$
$$j = idx \% q$$

arr2d =

	0	1	2	3	4	5
0	1	2	3	4	5	6
1	7	8	9	10	11	12

idx = 0, i = 0 j = 0
1, i = 0 j = 1
2, i = 0 j = 2
3, i = 0 j = 3
4, i = 0 j = 4
5, i = 0 j = 5
6, i = 1 j = 0
idx = 7, i = 1 j = 1
8 i = 1 j = 2
9 i = 1 j = 3
10 i = 1 j = 4
11 i = 1 j = 5

Note :- $idx = i * q + j$

from 2d to 1d

⇒ Binary Search

- BSLB
- BSUB

template :- array is always sorted

```
int i = 0;
int j = n-1;
while ( i <= j ) {
        int mid = (i+j)/2;
        if (arr[mid] == target){
            return mid;
        } else if (arr[mid] < target){
            i = mid+1;
        } else {
            j = mid-1;
        }
}
```

→ BSUB & BSLB

template :- array is always sorted

```
int i = 0;
int j = n-1;
while ( i <= j ){
    int mid = (i+j)/2;
    if (arr[mid] == target){
        return mid;
    } else if (arr[mid] < target){
        i = mid+1;
    } else {
        j = mid-1;
    }
}
```

(BSLB)

```
if( arr[mid-1] == arr[mid] ){
    j = mid-1;
} else {
    return mid;
}
```

(BSUB)

```
if( arr[mid+1] == arr[mid] ){
    i = mid+1;
} else {
    return mid;
}
```

→ Peak element

→ first & last occ. of target

→ Astroid collision

→ Max. Prod. subarray

→ Boat count

→ Search in sorted rotated array

- Transpose matrix
- Reach Target
- majority elements (leetcode)
- Smaller element on right
- Find Pivot index
- rotate array (leetcode)
- maximum subarray

→ Koko eating bananas

HW_Longest Palindromic Substring

13

str = " a b c b a a b c d e "

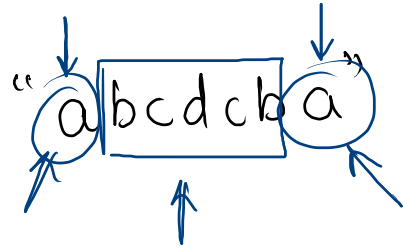
0 1 2 3 4 5 6 7 8 9

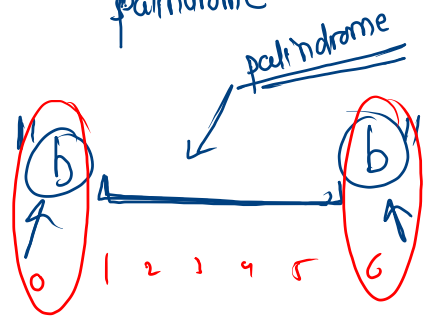
boolean

	0	1	2	3	4	5	6	7	8	9
0	T	F	F	F	T	F				
1		T	F	T	F	F	F			
2			T	F	F	F	F	T		
3				T	F	F	T	F		
4					T	T	F	F	F	
5						T	F	F	F	F
6							T	F	F	F
7								T	F	F
8									T	F
9										T

i = start of substring

j = end of substring





code

DP (Dynamic Programming)

```java
class Solution {
    public String longestPalindrome(String s) {
        int n = s.length(), count = 0, si = 0, ei = 0, len = 0;
        boolean[][] dp = new boolean[n][n];
        for (int g = 0; g < n; ++g) {
            for (int i = 0, j = g; j < n; ++i, ++j) {
                if (g == 0 || (g == 1 && s.charAt(i) == s.charAt(j)) || (s.charAt(i) == s.charAt(j) && dp[i + 1][j - 1])) {
                    dp[i][j] = true;
                    if (len < j - i + 1) {
                        si = i;
                        ei = j;
                        len = j - i + 1;
                    }
                }
            }
        }
        return s.substring(si, ei + 1);
    }
}
```