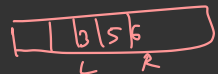
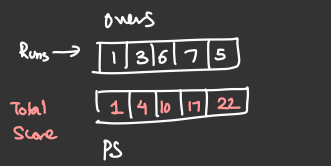
Some Questions → "Famous Algorithms" → invented after many attempts

Topic – Prefix Sums

Sum in Range (L, R)





ps[i] = ps[i-1] + arr[i];

Special Index (Assignment – Prefix Sums)

Given an array, arr[] of size N, the task is to find the count of array indices such that removing an element from these indices makes the sum of even-indexed and odd-indexed array elements equal.

one index at a time

Special

Brute Force

arr[6] = 4, 3, 2, 7, 6, -2

0 1 2 3 4 5

Remove Index 0:

3 2 7 6 -2

0 1 2 3 4

Copy of the Array

is a Spl index

$S_e = 8$

$S_o = 8$

Remove Index (1)

No

0	1	2	3	4
4	~~X~~	2	7	6	-2

$S_e = 9$
$S_o = 8$

Remove Index (2) :

Yes

0	1	2	3	4
4	3	~~X~~	7	6	-2

$S_e = 9$
$S_o = 9$

Remove Idx (3) :

No

0	1	2	3	4
4	3	2	~~X~~	6	-2

$S_o = 4$
$S_e = 9$

⋮

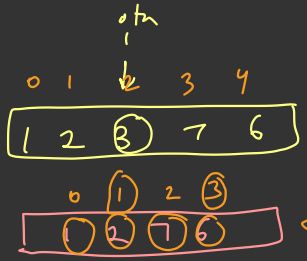
Count = 0

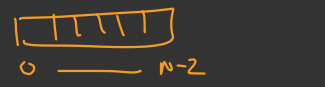
```
for (i=0 , i<=n-1 , i=i+1) {
    temp[] = removeIndex(arr, i);   // TODO // user created
}
```

N [temp[] ↑ elements (N-1)

$S_e = 0$, $S_o = 0$

Steps





$S_e = 1+7 = 8$

$S_o = 2+6 = 8$

Time = $O(N^2)$

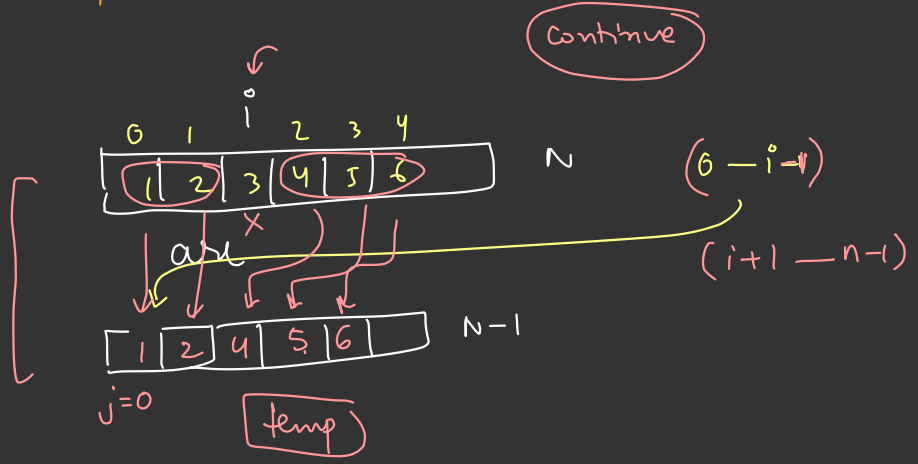
Space = $O(N)$

N

```
for (j = 0 ; j <= N-2 ; j = j+1){
    if (j is odd)   So = So + temp[j] ;
    else  j is even   Se = Se + temp[j];
}
if (Se == So){ cnt = cnt + 1 } ;
}
print (cnt)
```

Remove Idx

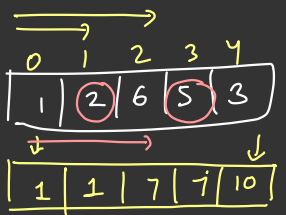
TODO $O(N)$



Prefix Even Sum

↓ Recap

ps[i] = store sum all even indice till i

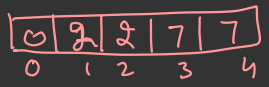


✓ [Prefix Sum]

$$ps[i] = \begin{cases} ps[i-1] & \text{// i is odd} \\ ps[i-1] + arr[i] & \text{// i is even} \end{cases}$$

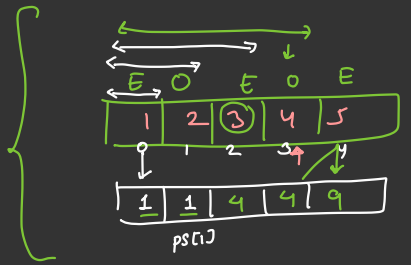
To Do: (handle for i == 0)

ps odd



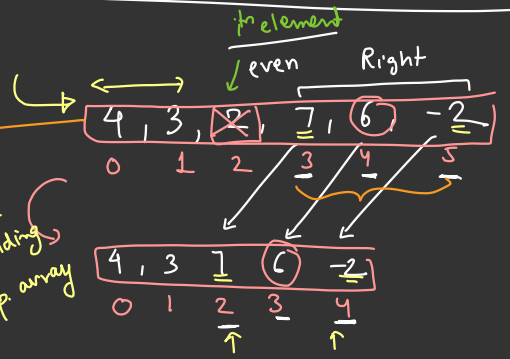
↳ Sum of all odd indices till i

[0,0] → No odd index (0)



Remove i^{th} element and compare the Sums

→ odd positions
→ even positions



odd → Even
Even → odd

1+1 ——— N-1
Toggled

$S_e = 4+7-2 = 9$

$S_o = 3+6 = 9$

⇒ without physically removing the element.

- $S_e = S_e[0\ 1] + S_o[3\ 5]$ → Converted into even positions
- $S_o = S_o[0\ 1] + S_e[3\ 5]$ → Converted into odd positions

Generalised

$S_e = S_e[0 \ldots i-1] + S_o[i+1 \ldots N-1]$

$S_o = S_o[0 \ldots i-1] + S_e[i+1 \ldots N-1]$

3 4

Result

Pre computed in O(N)

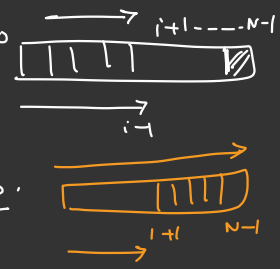
$S_e[0 \ldots i-1] = psEven[i-1]$ — $S_e = 0$ if $i == 0$

$S_o[i+1 \ldots N-1] = psOdd[N-1] - psOdd[i]$

$S_o[0 \ldots i-1] = psOdd[i-1]$ — $i == 0, S_o = 0$

$S_e[i+1 \ldots N-1] = psEven[N-1] - psEven[i]$

O(1) calc



Algo

step1 : psEven[N]
: psOdd[N] — O(N)

step2

```
for(i = 0 ____ n-1) {
    Se = ⟶ Result
    So =
    if (Se == So) { cnt = cnt + 1 }
}
```

O(N) time

O(N) space

Q Given N elements, find the majority element,

↳ freq of more than $\frac{N}{2}$ in the array

arr[6] = {1, 2, 1, 6, 1, 1}

Majority = 1 ✓ $\quad 4 > \frac{6}{2}$

arr[8] = {1, 4, 3, 2, 4, 4, 6, 4}

↳ No majority elements $\quad 4 \not> \frac{8}{2}$

Brute Force (1)

arr[1] → 1, 2, 1, 6, 1, 1

2

⋮

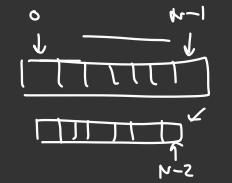
1 → 4

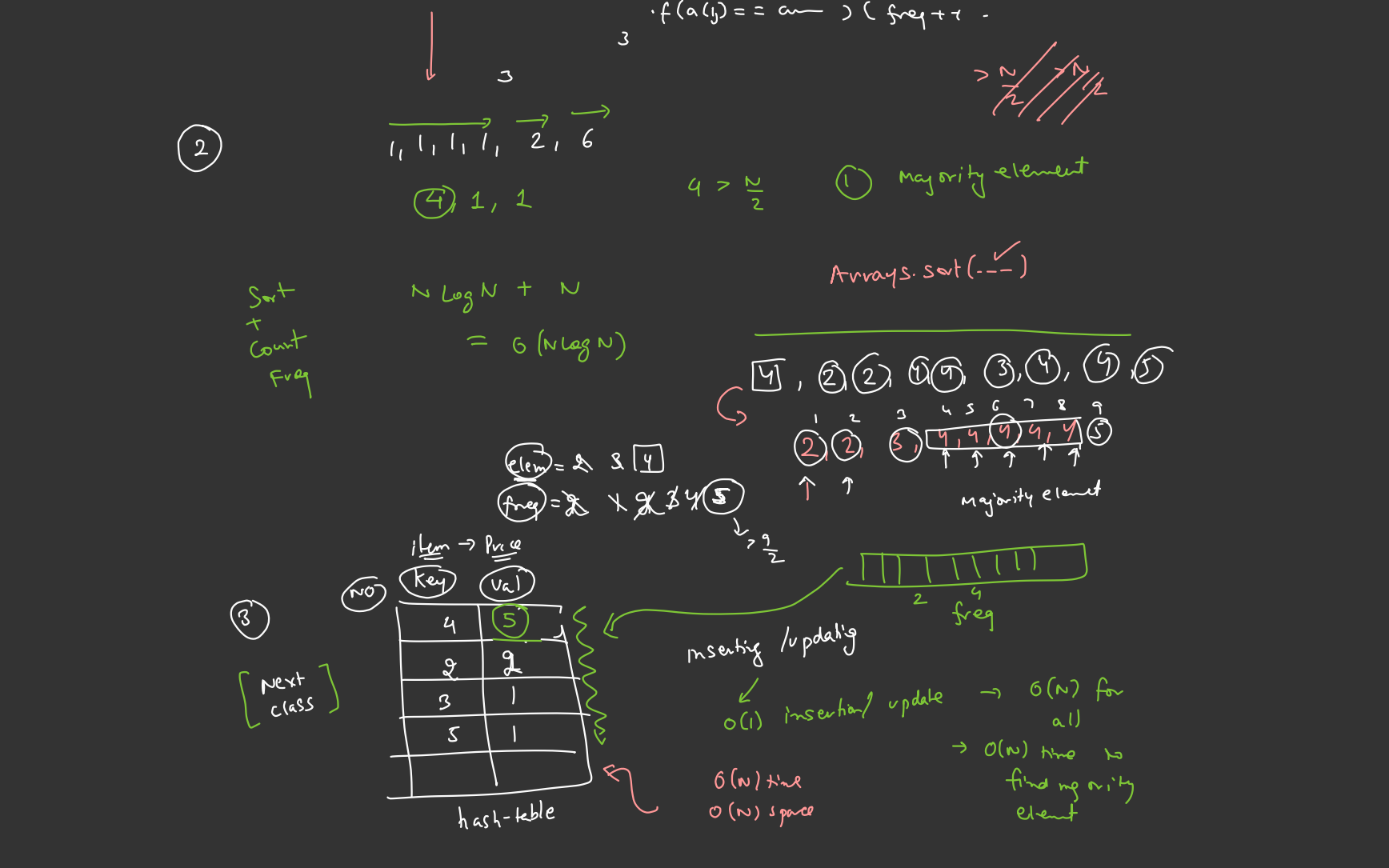
2 → 1

⋮

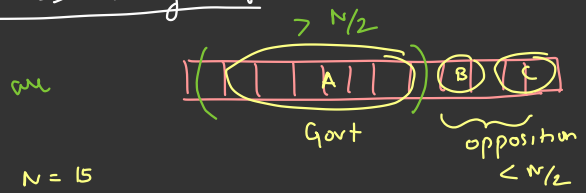
$O(N^2)$

```
for(i=0 ____ i<=N-1){
    // iterate & count freq of a[i]
    for(j=0 ____ j<=N-1){
```




if(a[i] == a[ ]) { freq++ ;
3
3
1, 1, 1, 1, 1, 2, 6
4 1, 1
> N/2
2
1 Majority element
Sort + Count Freq
N Log N + N
= O(N Log N)
Arrays.sort(---)
4, 2, 2, 4, 4, 3, 4, 4, 5
1 2 3 4 5 6 7 8 9
2, 2, 3, 4, 4, 4, 4, 4, 5
Majority element
elem = 2 & 4
freq = 2 X 2 3 4 5
> 9/2
Item → Price
NO
Key
Val
4 5
2 2
3 1
5 1
hash-table
3
[Next class]
2
4
freq
inserting / updating
O(1) insertion/ update → O(N) for all
→ O(N) time to find majority element
O(N) time
O(N) space

4 Moore's Voting Algo

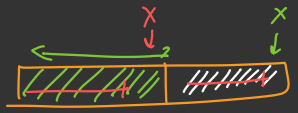


N = 15

- Party A – 
- Party B – 
- Party C – 

9 + 2 + 4 = 15

Disqualify 2 candidates belonging to diff parties

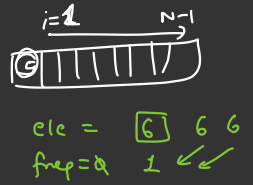


15 → 9 seats
13 → 8 seats
11 → 7 seats
9 → 7 seat
7 → 6 seat
5 → 5 seats


Ex-1
0 1 2 3 4 5 6 7 8 9
4, 6, 5, 3, 4, 5, 6, 4, 4, 4
5
11
6, 13
Yes
Element → 4 5 4 6 4
freq → 1 0 1 0 1 0 1 0 1 2
need to check the actual freq of element in arr
may be
4, 6
5, 3
4, 5
6, 4
4, 4
Ex-2
ele
arr(i)
3, 4, 3, 3, 6, 3, 7
element = 3
freq = 1 0 1 2 2 1
4 / 7
3, 4
3, 3
3, 6
Check the freq of element > n/2
4 → 3S 4 > n/2
3 is majority element
Ex-3
3, 3, 4, 3, 4, 4, 1
ele = 3
1
f = 2 2 1 0 1
freq = 1 ?
freq(1) > 7/2

Code

```
int majority ( int arr[] ) {
    N = arr.size()
    [ element = arr[0]
      freq = 1
    for (i = 1 ——— N-1) {
        if ( freq == 0 ){
            element = a[i]
            freq = 1
        }
        else if ( element == arr[i]){
            freq += 1
        }else { // 2 diff elem }
            freq = freq - 1
    }
// go and count freq of element
    cnt = 0
    for (i = 0 , i <= N-1 , i++) {
        if ( arr[i] == element ) { cnt++ }.
    }  if (cnt > N/2) { element }
```



Magic No Nth Magic No

5, 25, 30, 125, 130, 155 ----

Question

5 → 5^1

25 → 5^2

30 → $5^2 + 5^1$

125 → 5^3

130 → $5^3 + 5^1$

150 → $5^3 + 5^2$

155 → $5^3 + 5^2 + 5^1$

Nth magic

Base 2 | Base 5

0 → 0 0 0 | 0

1 → 0 0 1 | $5^3\ 5^2\ 5^1$ → 001 → 5

2 → 0 1 0 | 010 → 25

3 → 0 1 1 | → 30

4 → 1 0 0 | → 125

5 → 1 0 1 | → 130

6 → 1 1 0 | → 150

$5^3\ 5^2\ 5^1$

1 0 0

$5^3\ 5^2\ 5^1$

1 1 0

6 Magic NO →

$= 5^3 \times 1 + 5^2 \times 1 + 5^1 \times 0$

$= 125 + 25 + 0$

$= 150$

6 → 1 1 0 (powers $5^2\ 5^1\ 5^0$)

$0 + 1 \times 5^2 + 1 \times 5^1$

$= 25 + 125$

$= 150$

```
p = 5
while (N > 0) {
    last_bit = N & 1
    ans += last_bit * p;
    N = N >> 1
    p = p * 5
}
print(ans);
```

Q Largest NO

hint

3 , 43 , 26

a + b > b + a

3 < 26

3,43,26

43,26,3

43,3,26

26 > 3

detail

a b

3 , 43

3 + 43 = 343

43 + 3 = 433

a + b > b + a

(join) (join)

3 , 43 , 26

=>

if (compare (a, b))

4

43, 3, 26

43 3 26

3 + 26 = 326 < 26 + 3 = 263

Compare (string a, string b) {

a + b > b + a

=

Q Josephus Problem (Puzzle)

N people in circle, Person 1 has Knife

he kills adj person and passes Knife to next adjacent person. Repeat!

clockwise

Who is the last person

Observations

Example

N = 7

ans = 7

× Straightforward Powers of 2

N = 2

Winner 1

N = 4

1

N = 8

1

N = 16

1

N = 32

1

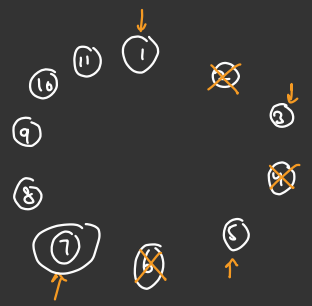
↳ for N that is power of 2, person who starts the game wins it

Thm

N = 11 (not a power of 2)

11 − 3 = 8

3



Kill (3) people

↓

People are left → 8 ppl

⇑

Power of 2

(7) will win game

Formula

Kill = N − nearest Power of 2

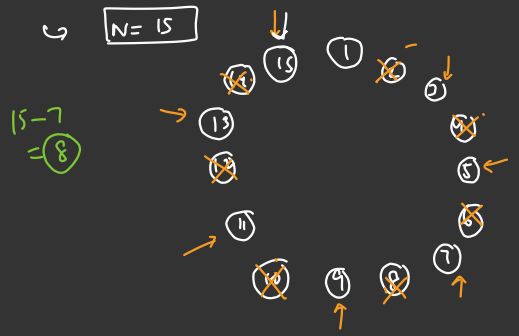
Output 1 = 2(Kill) + 1

= 2(11 − 8) + 1

= 2(3) + 1

= (1)

↳ N = 15



15 − 7 = (8)

Kill = (7)

15 − 7 = (8)

→ 15 will win the game

N = 15 → 8

Kill = 7

Winner = 2Kill + 1

= 2(7) + 1

= (15)

N = 100

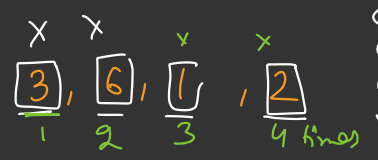
Kill = 100 − 64

= 36

Output = 2(36) + 1

= 73 ✓

Q



Cost = (6+1+2+3) = 12
cost = (1+2+6) = 9
cost = (2+1) = 3
Cost = (2) = 2
26 ↓ minimise cost

Given an integer array A of size N. You can remove any element from the array in one operation.
The cost of this operation is the sum of all elements in the array present before this operation. Find the minimum cost to remove all elements from the array.

Cost = arr[i] → 1 times

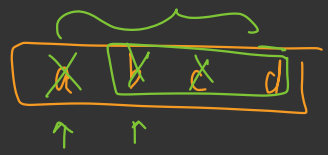
3, 6, 1, 2

= 3(1) + (6)(2) + 1(3) + 2(4)
= 3 + 12 + 3 + 8 = 26

6, 3, 2, 1
×1 ×2 ×3 ×4

6 + 6 + 6 + 4

= 22



$c_1 = a + b + c + d$

$c_2 = b + c + d$

$c_3 = c + d$

$c_d = d$

Cost $4d + 3c + 2b + 1a$