Q Given N elements and Q queries. For each query calculate sum of elements from L to R inclusive. (0 based index).

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| -3 | 6 | 2 | 4 | 5 | 2 | 8 | -9 | 3 | 1 |

A = [-3, 6, 2, 4, 5, 2, 8, -9, 3, 1]

Queris

4 8 = 9

3 7 = 10

1 3 = 12

0 9 = 19

7 7 = -9

Brute Force: For each query, iterate from L to R and find sum.

```
query Sum (queries[][], arr) {
    for(int i=0; i<queries.size(); i++) {
        L = queries[i][0]
        R = queries[i][1]

        sum = 0
        for(int j = L; j<=R; j++) {
            sum += Arr[j]
        }
        print(sum)
    }
}
```

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

$N = 10^5$  $Q = 10^5$

Worst Case No. of iterations : $10^{10}$

$\downarrow\downarrow$

100 sec.

Pass?

$\downarrow\downarrow$

NO

Quiz 1: Given the scores of 10 overs of a cricket match :

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 8 | 14 | 29 | 31 | 49 | 65 | 79 | 88 | 97 |

Find score in $7^{th}$ over.

Ans: Score after $7^{th}$ overs − Score before $7^{th}$ over
(Score after $6^{th}$ overs)

= 65 − 49 = 16

Quiz 2: Find score from $6^{th}$ to $10^{th}$ over ?
(both inclusive)

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 8 | 14 | 29 | 31 | 49 | 65 | 79 | 88 | 97 |

Ans: Score till $10^{th}$ over − Score before $6^{th}$ over
(Score after $5^{th}$ over)

97 − 31 = 66

Quiz 3:

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 8 | 14 | 29 | 31 | 49 | 65 | 79 | 88 | 97 |

Find score in just 10th over?

Ans: Score till 10th over - Score before 10th over
(Score after 9th over)

$97 - 88 = 9$

Quiz 4:

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 8 | 14 | 29 | 31 | 49 | 65 | 79 | 88 | 97 |

Find score from 3rd to 6th over?
(both inclusive)

Ans: Score till 6th over - Score before 3rd over
(Score till 2nd over)

$49 - 8$

$= 41$

Quiz 4:

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 8 | 14 | 29 | 31 | 49 | 65 | 79 | 88 | 97 |

Find score from 4th to 9th over?
(both inclusive)

Ans: Score till 9th over - Score before 4th over
(Score till 3rd over)

$S[9] - S[3]$

$= 88 - 14 = 74$

Observations :

→ If we have cumulative sum then answering queries faster.

→ Similarly If we make a cumulative sum array for our 1st problem

⇒ It will simplify answering our queries aswell

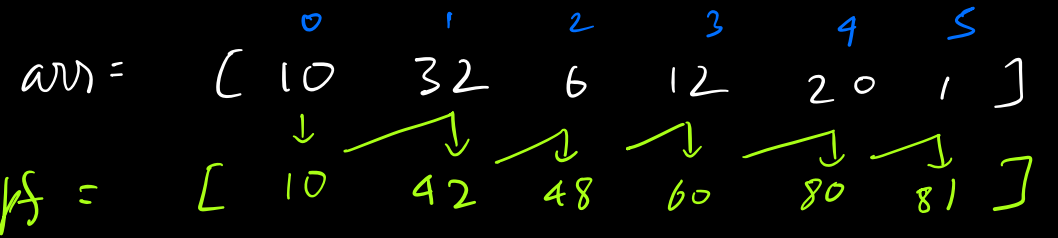
The cumulative sum Array = Prefix Sum (The scoreboard type Array)

How to create a prefix Array?

ps[i] = Sum of all the elements from 0 to i

```
      0   1   2   3   4
arr = [2   5  -1   7   1 ]
       ↓   ↓   ↓   ↓   ↓
ps  = [2   7   6  13  14 ]
```

# Quiz 6: Calculate prefix sum array:




Brute force Way:

```
pf[N]
for (i = 0; i < N; i++) {
    sum = 0
    for (int j = 0; j <= i; j++) {
        sum += arr[j]
    }
    pf[i] = sum;
}
```

$pf[0] = A[0]$

$pf[1] = A[0] + A[1]$

$pf[2] = A[0] + A[1] + A[2]$

$pf[0] = A[0]$

$pf[1] = pf[0] + A[1]$

$pf[2] = pf[1] + A[2]$

$\Longrightarrow pf[i] = pf[i-1] + A[i]$

```
pf[N]
pf[0] = A[0];
for(i = 1; i<N; i++){
    pf[i] = pf[i-1] + A[i]
}
```

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

Q How to answer queries using Prefix Array.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| A = | -3 | 6 | 2 | 4 | 5 | 2 | 8 | -9 | 3 | 1 |
| pf = | -3 | 3 | 5 | 9 | 14 | 16 | 24 | 15 | 18 | 19 |

Queris

4 8 = pf[8] - pf[3] = 18 - 9 = 9

3 7 = pf[7] - pf[2] = 15 - 5 = 10

1 3 = pf[3] - pf[0] = 9 - (-3) = 12

0 4 = pf[4] = 14

7 7 = pf[7] - pf[6] = 15 - 24 = -9

sum[L R] = pf[R] - pf[L-1]

if L == 0, sum[0 R] = pf[R]

```
querySum(queries[][], arr) {
    # Create Prefix Array
    pf[N]
    pf[0] = A[0];
    for(i = 1; i < N; i++) {
        pf[i] = pf[i-1] + A[i]
    }
    # Answer Queries
    for(int i=0; i < queries.size(), i++) {
        L = queries[i][0]
        R = queries[i][1]
        if(L == 0) {
            sum = pf[R]
        }else{
            sum = pf[R] - pf[L-1]
        }
        print(sum)
    }
}
```

TC: $O(N+Q)$ SC: $O(N)$

Space Optimised can be further by modifying input array.

```
void convertToPrefixArray( arr[] ){
    for(i=1; i<arr.size(); i++){
        Arr[i] = Arr[i-1] + Arr[i];
    }
}
```

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

Q Given an array of size N and Q queries. Query is given as (L, R), return the sum of even indexed elements from L to R.

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| A = [ | 2 | 3 | 1 | 6 | 4 | 5 ] |

Query

1 3 => A[2] = 1

2 5 => A[2] + A[4] = 5

0 4 => A[0] + A[2] + A[4] = 7

3 3 => — = 0

Brute Force: Iterate in L to R and sum even index for each query.

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

## Observation to Optimise

→ We need to only consider even indices while making prefix array

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| A = [ | 2 | 3 | 1 | 6 | 4 | 5 ] |
| pSE = [ | 2 | 2 | 3 | 3 | 7 | 7 ] |

pSE[i] = Sum of all even index from 0 to i

if i is even :

pSE[i] = pSE[i-1] + A[i]

else (i is odd):

pSE[i] = pSE[i-1]

Quiz 7: Create Prefix Sum Even for the below array

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| [ | 2 | 4 | 3 | 1 | 5 ] |
| pSE [ | 2 | 2 | 5 | 5 | 10 ] |

```
void SumOfEvenIndex(int A[], int queries[][], int N){
    // Create Prefix Array
    int pSE[N];
    pSE[0] = A[0]
    for(i = 1; i < N; i++){
        if(i % 2 == 0){
            pSE[i] = pSE[i-1] + A[i]
        }
        else{
            pSE[i] = pSE[i-1]
        }
    }
    // Answer Queries
    for(int i=0; i<queries.size(), i++){
        L = queries[i][0]
        R = queries[i][1]
        if(L == 0){
            sum = pf[R]
        }
        else{
            sum = pf[R] - pf[L-1]
        }
        print(sum)
    }
}
```

TC: O(N + Q)  SC: O(N)

Extension Problem:

Find the sum of odd indexed elements?

Change the prefix Array.

```
pso[N]

pso[0] = 0;
if i%2 == 1:
    pso[i] = pso[i-1] + A[i]
else
    pso[i] = pso[i-1]
```

## Special Index

Given an array of size N, count the number no. of special index in the array.

After removing a special index from the array: Sum of all EVEN index = Sum of all ODD index

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| A[] = [ | 4 | 3 | 2 | 7 | 6 | -2 ] |

| i | A[] | Se | So | |
|---|---|---|---|---|
| 0 | 3 2 7 6 -2 | 8 | 8 | = ✓ |
| 1 | 4 2 7 6 -2 | 9 | 8 | = X |
| 2 | 4 3 7 6 -2 | 9 | 9 | = ✓ |
| 3 | 4 3 2 6 -2 | 4 | 9 | = X |
| 4 | 4 3 2 7 -2 | 4 | 10 | = X |
| 5 | 4 3 2 7 6 | 12 | 10 | = X |

ans = 2

Quiz What will be the sum of elements at ODD indices in the resulting array after removal of index 2?

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| A[] = [ | 4 | 1 | 3 | 7 | 10 ] |

4 1 7 10

Ans = 1 + 10 = 11

|| pSO[0, i-1] || pSE[i+1 N-1]

Quiz:

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| A = | 2 | 3 | 1 | 4 | 0 | -1 | 2 | -2 | 10 | 8 |
| pso = | 0 | 3 | 3 | 7 | 7 | 6 | 6 | 4 | 4 | 12 |
| pse = | 2 | 2 | 3 | 3 | 3 | 3 | 5 | 5 | 15 | 15 |

Deleting index 3

Sum of All ODD after removing index 3

= pso[2] + (sum of even in 4 to 9)

+ pse[9] − pse[3]

= 3 + 15 − 3 = 15

Quiz Sum of Even index after removing index 3.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| A = | 2 | 3 | 1 | 4 | 0 | -1 | 2 | -2 | 10 | 8 |
| pso = | 0 | 3 | 3 | 7 | 7 | 6 | 6 | 4 | 4 | 12 |
| pse = | 2 | 2 | 3 | 3 | 3 | 3 | 5 | 5 | 15 | 15 |

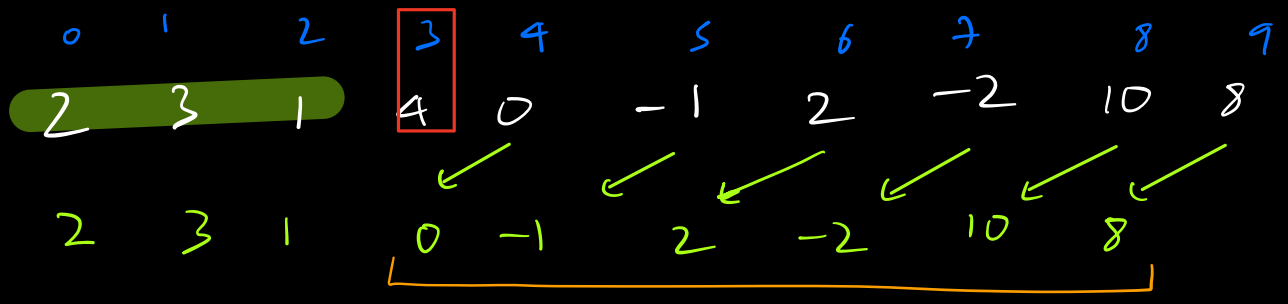
→ Array before i in unchanged

→ Array after i odd ⇒ Even
Even ⇒ Odd

Sum of all EVEN after removing 3 = pse[2] + Sum of odd from 4 to 9

pso[9] − pso[3]

= 3 + 12 − 7 = 8




All odd indices becomes Even and even becomes odd

After removal of $i^{th}$

Sum of Even = Sum of Even till $i-1$

+

Sum of ODD from $i+1$ to $N-1$

$= pSE[i-1] + pSO[N-1] - pSO[i]$

Sum of Odd = Sum of ODD till $i-1$

+

Sum of EVEN from $i+1$ to $N-1$

$= pSO[i-1] + pSE[N-1] - pSE[i]$

```
int countSpecialIndex(int arr[], int n)

    int PSE[N]
    int PSO[N]

    # TODO FILL PSE and PSO

    int count = 0;
    for(i = 0; i < N; i++) {    // Check for removal of i

        int SE, SO;

        if(i == 0) {
            SE = pso[N-1] - pso[i]    //SUM(0 to N-1)
            SO = pse[N-1] - pse[i]    //SUM(0 to N-1)
        }
        else {
            SE = pse[i-1] + pso[N-1] - pso[i]
            SO = pso[i-1] + pse[N-1] - pse[i]
        }

        if(SE == SO){
            count++;
        }

    }
    return count
}
```

TC: $O(2N \text{ or } 3N) \approx O(N)$

SC: $O(2*N) \approx O(N)$

Next Class Topic:

1) Carry Forward Technique
   → It's a clever way of passing already computed info ahead in the array

2) Basics of Subarray
   → What ?
   → where ?
   → Some nuances in problems

Subarray is very very common term used in lots of array problems.

Story:
→ Vidya's applied for Amazon
→ She got OA
   2 questions, 1 hr.
→ 101 questions.

→ Coded it up but got TLE = Time Limit Exceeded

→ She got optimation ided

→ She again coded it up TLE

→ She again got more optimisation idea

→ She again coded it up ☺ Passed

10 sec. left

No. of iteration ⇒ TC

But values of N, Q . . . .

$10^8$ iteration = 1 sec.

How much time your code will take.

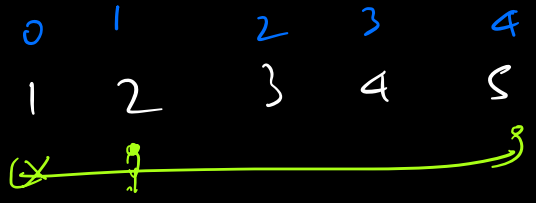
$1 <= N <= 10^5$

| Complexity | Iterations | Works. |
|---|---|---|
| $O(N^3)$ | $10^{15}$ | ✗ |
| $O(N^2) \log N$ | $\simeq 20 * 10^{10}$ | ✗ |
| $O(N^2)$ | $10^{10}$ | ✗ |
| $O(N * \log N)$ | $\simeq 20 * 10^5$ | ✓ |

$1 \le N \le 10^6$

| Complexity | Iterations | Works. |
|---|---|---|
| $O(N^3)$ | $10^{18}$ | ✗ |
| $O(N^2) \log N$ | $\simeq 23 \times 10^{12}$ | ✗ |
| $O(N^2)$ | $10^{12}$ | ✗ |
| $O(N * \log N)$ | $23 \times 10^6$ $= 2.3 \times 10^7$ | May be (Mostly yes but may be) |
| $O(N)$ | $10^6$ | ✓ |

## Doubts




0 → [2 3 4 5] ✗ again in sum ~ ODD

1 → [1 3 4 5] — — — Sum ...

TC: $O(N^2)$ SC: $O(N)$

$N = 10^5 \Rightarrow$ $10^{10}$ iterations

100 sec.

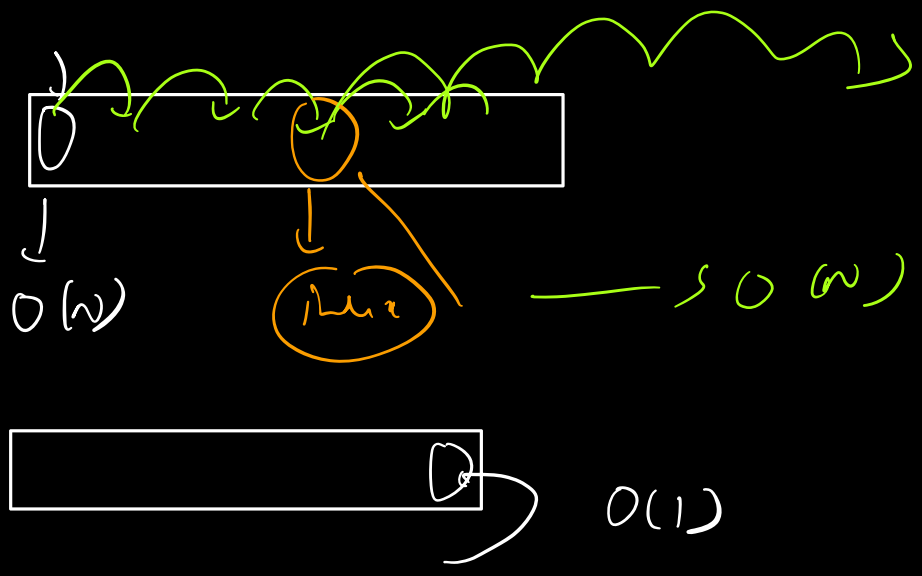
$2^{10} = 1024 \simeq 1000$

$10^3 \simeq 2^{10}$

$10^6 \simeq 2^{20}$

$\log(10^6) \quad \log(2^{20})$

$= 20$




Try to make examples

To run through it

Try to draw out some observations

20 min only