Agenda

1. Rotate Array K times
2. Prefix Sum Introduction
3. Modify a given array to Prefix Array
4. Sum of Even Indexed Elements in a given Range
5. Sum of Odd Indexed Elements in a given Range
6. Special Index

Rotate Array K times

Given an array 'arr' of size 'N'. Rotate the array from right to left 'K' times. (i.e, if K = 1, last element will come at first position,...)

TestCase:

Input:

```
N = 5
arr = {1,2,3,4,5}
k = 2
```

0 1 2 3 4

$\Rightarrow$ K = 1 $\Rightarrow$ { 5, 1, 2, 3, 4 }

Output:

```
arr = {4,5,1,2,3}
```

0 1 2 3 4

I/P $\Rightarrow$ Static Array.

Soln : Brute Force Approach

One rotation

(N-2) (N-1)

temp = A[N-1]

A = { 1, 2, 3, 4, 5 }

0 1 2 3 4

5 1 2 3 4

⇒ Perform K rotations one by one.

Code

```
void rotateArray (int A[], int N, int K){
    K = K % N;
    for (i = 0; i < K; i++){ // Rotation K times.
        temp = A[N-1];
        for (j = (N-2); j >= 0; j--){
            A[j+1] = A[j];
        }
        A[0] = temp;
    }
}
```

(N-1) → N

} Shifting

T.C. = O(N × K)

S.C. = O(1)

A = { 1, 2, 3, 4, 5 } (0, 1, 2, 3, 4)

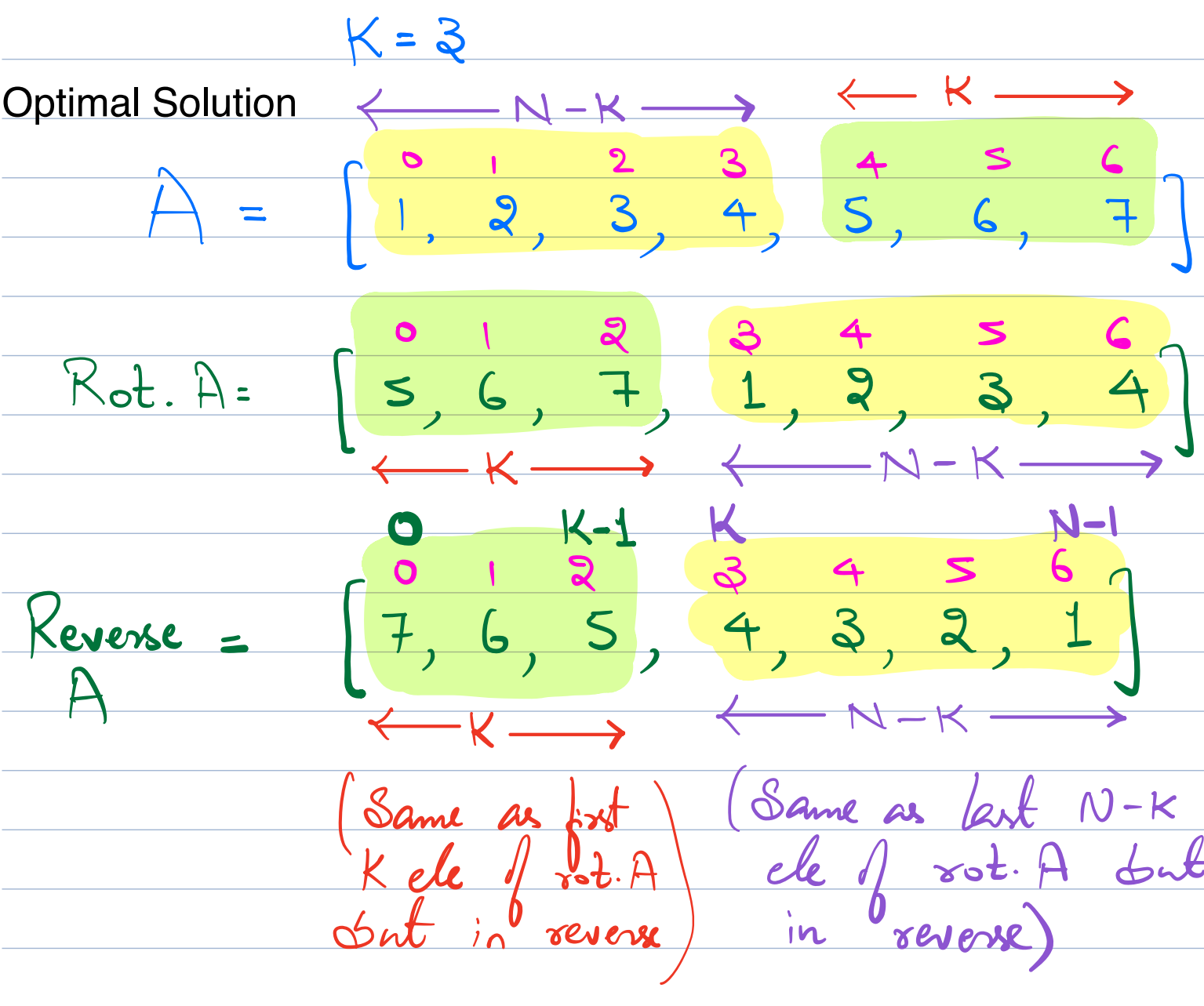
K = 5 (Original array)

K = 10

K = 12 ⇒ 2 (K % N)

5

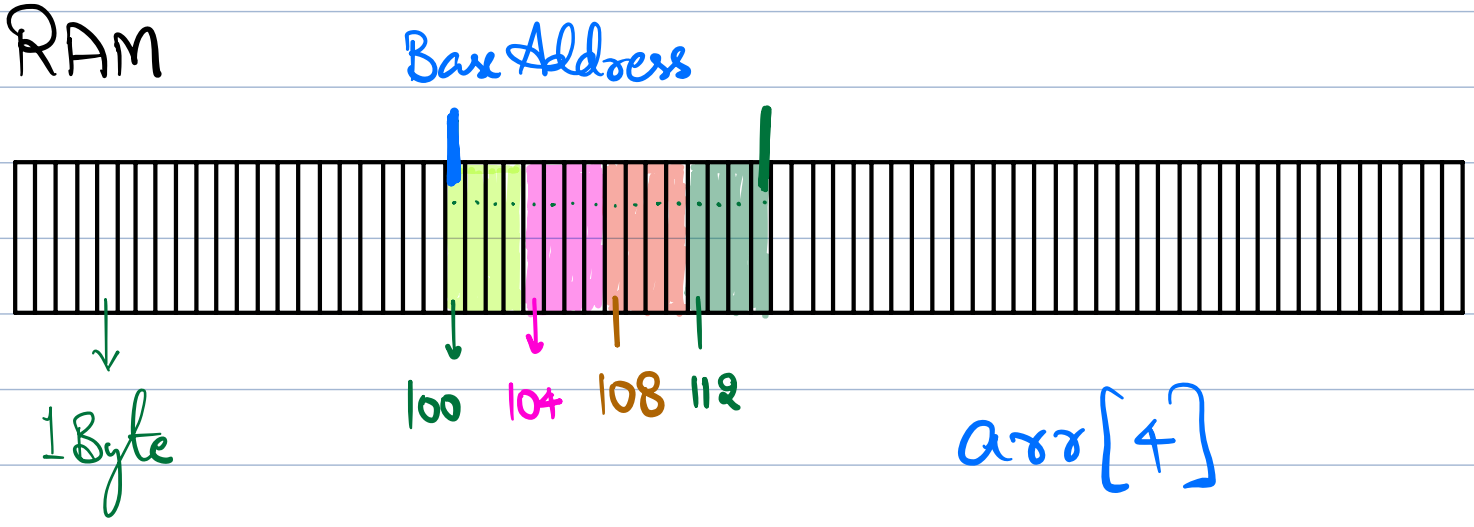
Optimal Solution



Code

1) reverse (A, N, 0, N-1); O(N)
2) reverse (A, N, 0, K-1); + O(K)
3) reverse (A, N, K, N-1); + O(N-K)

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



Dynamic Arrays

Static array but with automatic resizing.

```
Arraylist<int> arr = new Arraylist<int>(10);
```

10 → size.

Default value.

10 (Static array) (Capacity)

Add elements

1
2
⋮
7
8 (80%)

Load factor = 0.75

> 75%.

new static array of double size (20)

Revision Quiz 1

Find the Space Complexity [Big(O)] of the below program.

```
func(int N) { // 4 bytes
    int arr[10]; // 40 Bytes
    int x; // 4 bytes
    int y; // 4 bytes
    long z; // 8 bytes
    int arr[N]; // 4 * N bytes
}
```

~~60~~ + ~~4~~N

= O(N)

Revision Quiz 2

What is the time complexity of accessing element at the ith index in an array of size N?

4 options

Active Duration (Most preferred: 30 seconds)

Appears for	30 Secs

A	O(N)
B	O(1)
C	O(N*N)
D	O(log(N)).

Revision Quiz 3

What does the following Pseudocode do ?

```
Function(arr[], N){
    i=0, j=N-1;
    while(i < j) {
        temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
        i++;
        j--;
    }
}
```

⇒ Reverse

Stock Portfolio Performance Tracking

Problem :

Tracking the performance of stocks over time is crucial for making informed decisions. You want to develop a feature for a banking app that allows users to quickly assess their stock portfolio's profit or loss over specified periods. To efficiently calculate the total profit or loss over any given range of time, you decide to implement this feature using the prefix sums technique.

Problem Statement :

Given an array representing the daily profit or loss from a particular stock over a period of days, write a function that calculates the total profit or loss over a given range of days. The function should efficiently handle multiple queries for different ranges without recalculating the sum for each query.

Example:

0 1 2 3 4 5 6 7 8 9

Stock_Prices[ ] = [-5, 10, 20, 40, 50, -10, 80, -90, -20, -10]

Queries (Q)

Range Queries

Start Day	End day	Net Stock Price
0	9	65
1	4	120
0	0	-5
7	9	-120
2	7	90

The problem is same as calculating sum of elements within a given range. Let's explore it further.

Range Sum Query

Problem Description

Given N elements and Q queries. For each query, calculate sum of all elements from L to R [0 based index].

$1 \le N \le 10^5$

$1 \le Q \le 10^5$

Example:

0 1 2 3 4 5 6 7 8 9

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

Queries

Queries [0] [2]

L	R	
4	8	→ 9
3	7	→ 10
1	3	→ 12
0	4	→ 14
7	7	→ -9

Brute Force Approach

For each query, iterate over the range & calculate the sum.

Code

```
function querySum(int A[], int N, int queries[][],
                  int Q) {

    for (i = 0; i < Q; i++) {          // Q

        L = queries[i][0];
        R = queries[i][1];
        Sum = 0;
        for (j = L; j <= R; j++) {     // [L, R]   ↑ N

            Sum = Sum + A[j];
        }

        print (Sum);
    }
}
```

$T.C. = O(Q \times N)$ $\quad 10^5 \times 10^5 = 10^{10}$??

$S.C. = O(1)$

Optimal Solution

Cricket

Given the score of the team after each over.

QUIZ 1

Given the scores of the 10 overs of a cricket match
2, 8, 14, 29, 31, 49, 65, 79, 88, 97
(1 2 3 4 5 6 7 8 9 10)

How many runs were scored in just 7th over?

Score [7] − Score [6]

65 − 49 = 16

Given the scores of the 10 overs of a cricket match
2, 8, 14, 29, 31, 49, 65, 79, 88, 97
(1 2 3 4 5 6 7 8 9 10)

How many runs were scored from 6th to 10th over(both included)?

Score [10] − Score [5]

97 − 31

= 66

Given the scores of the 10 overs of a cricket match

2, 8, 14, 29, 31, 49, 65, 79, 88, 97

How many runs were scored in just 10th over?

$$Score[10] - Score[9]$$

$$97 - 88 = \underline{\underline{9}}$$

Given the scores of the 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

How many runs were scored from 3rd to 6th over(both included)?

Score[6] − Score[2]

49 − 8

= 41

2, 6, 6, 15, 2, 18, 16, 14, 9, 9

Given the scores of the 10 overs of a cricket match

2, 8, 14, 29, 31, 49, 65, 79, 88, 97

How many runs were scored from 4th to 9th over(both included)?

Score[9] − Score[3]

88 − 14

= 74

Runs scored from i^{th} to j^{th} over (Both inclusive) = Score[j] − Score[i − 1]

Observation for Optimised Solution

⇒ Cumulative (Sum) runs scored after each

over.

⇒ Total score of team after each over

How to create Prefix Sum Array

⇒ If cumulative data is given. Range queries can be answered very efficiently.

⇒ Prefix Sum Array

↓

from the start

Prefix[i] ⇒ Sum of elements from index 0 to index i (Both inclusive)

$A = [\overset{0}{2}, \overset{1}{5}, \overset{2}{-1}, \overset{3}{7}, \overset{4}{1}]$

$PS = [2, 7, 6, 13, 14]$

$PS[0] = Sum[0,0] = A[0]$

$PS[1] = Sum[0,1] = A[0] + A[1]$

$PS[2] = Sum[0,2] = \underbrace{A[0] + A[1]}_{PS[1]} + A[2]$

$PS[3] = Sum[0,3] = A[0] + A[1] + A[2] + A[3]$

$PS[2]$

$PS[i] = PS[i-1] + A[i]$

Prefix Sum Array Calculation

Calculate the prefix sum array for following array:-

10 32 6 12 20 1

A ⇒ I/P Array
N ⇒ Size

```
PS[N];

PS[0] = A[0];

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

T.C. = O(N)

How to answer the Range Sum Queries using Prefix Array

$$Sum[L,R] = PS[R] - PS[L-1]$$

```
If (L=0) {
    Sum[0,R] = PS[R]
}
```

Code

```
function querySum( int A[], int N, int queries[][],
                   int Q) {
// Calculate PS array.                    // N
for (i = 0; i < Q; i++) {                 // Q
    L = queries[i][0];
    R = queries[i][1];
    Sum;
    if (L == 0) { Sum = PS[R]; }
    else { Sum = PS[R] - PS[L-1]; }

    print (Sum);
  }
}
```

O(1)

T.C. = O(N + Q)

S.C. = O(N)

↳ PS Array.