Welcome 

Agenda : Prefix Sum
2-3 problems

Quiz

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

---

Q

eg:

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| -5 | 10 | 20 | 40 | 50 | -10 | 80 | -50 | -20 | -10 |

Queries

| Start day | End day | Net Stock price. |
|---|---|---|
| 0 | 9 | +65 |
| 1 | 4 | +120 |
| 0 | 0 | -5 |

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

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

Queries (Q)

[4, 8] → 9

Brute force

for each query, we iterate from L to R and sum the elements from L to R

Pseudocode.

```
for ( i=0 ; i< queries.length() ; i++)
{
    L = queries[i][0]
    R = queries[i][1]
    sum = 0
    for ( j= L ; j ≤ R ; j++)
    {
        sum += A[j]
    }
    print ( sum)
}
```

T.C ⇒ N * Q

S.C ⇒ O(1)

2D Array.length() → # of rows.

2D Array[0].length() → # of columns.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 8 | 14 | 23 | 31 | 49 | 65 | 79 | 88 | 97 |

7th over? $\Rightarrow$ $S[7] - S[6] = 65 - 49 = 16$

$[6,10]$ $\Rightarrow$ $S[10] - S[6] - S[5] = 97 - 31 = 66$

$[10,10]$ $\Rightarrow$ $S[10] - S[9] = 9$

$\Rightarrow$ If we have cumulative sum of array for the above problem, we should be able to answer queries in constant time.

How to create prefix sum array

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| eg: A : | 2 | 5 | -1 | 7 | 1 |
| Psum : | 2 | 7 | 6 | 13 | 14 |

Quiz

| | | | | | | |
|---|---|---|---|---|---|---|
| A : | 10 | 32 | 6 | 12 | 20 | 1 |
| Psum : | 10 | 42 | 48 | 60 | 80 | 81 |

Brute force

```
Psum[N]
for( i=0 ; i<N ; i++)
{  sum=0
   for( j=0 ; j<= i ; j++)
   {  sum += A[j]
```

```
    }
    Psum[i] = sum
}
```

T.C = O(N*N)
O(N²)

Obs

Psum[0] = A[0]

Psum[1] = A[0] + A[1]

Psum[2] = A[2] + A[1] + A[0]

Reuse

Psum[0] = A[0]

Psum[1] = A[1] + Psum[0]

Psum[2] = A[2] + Psum[1]

⋮

| Psum[i] = A[i] + Psum[i-1] |
|---|

Optimise code

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

T.C ⇒ O(N)

S.C ⇒ O(N)

1. Create prefix sum array.

```
for ( i=0 ; i< queries.length() ; i++)
{
    L = queries[i][0]
    R = queries[i][1]
    if ( L == 0)
        sum = Psum[R]
    else
        sum = Psum[R] - Psum[L-1]
}
```

T.C ⇒ ~~O(N*Q)~~

⇒ O(N+Q)

S.C ⇒ O(N)

Optimize space complexity

Drawback → original array will be lost.

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

T.C ⇒ O(N)

S.C ⇒ O(1)

Q Given an array of size N and Q queries with start (s) & end (e) index. For each query, return sum of all even indexed elements from s to e.

eg:

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| A : | 2 | 3 → 0 | 1 | 6 → 0 | 4 | 5 → 0 |
| Psum$_e$ | 2 | 2 | 3 | 3 | 7 | 7 |

Query

| | | |
|---|---|---|
| 1 | 3 | → 1 |
| 2 | 5 | → 5 |
| 0 | 4 | → 7 |
| 3 | 3 | → 0 |

Brute force

for each query, iterate over array and add it to sum if it is even indexed element.

TC ⇒ O(N * Q)

Optimization

Even index

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

Odd index

$$Psum_o[i] = Psum_e[i-1]$$

Code

$Psum_e[N]$ $Psum_e[0] = A[0]$

1. Create prefix sum for even index.

```
for( i = 1 ; i < N ; i++)
{
    if( i % 2 == 0)
        Psum_e[i] = Psum_e[i-1] + A[i]
    else
        Psum_e[i] = Psum_e[i-1]
}
```

O(N)

```
for( i = 0 ; i < queries.length() ; i++)
{
    L = queries[i][0]
    R = queries[i][1]
    if( L == 0)
        sum = Psum_e[R]
    else
        sum = Psum_e[R] - Psum_e[L-1]
    print( sum)
}
```

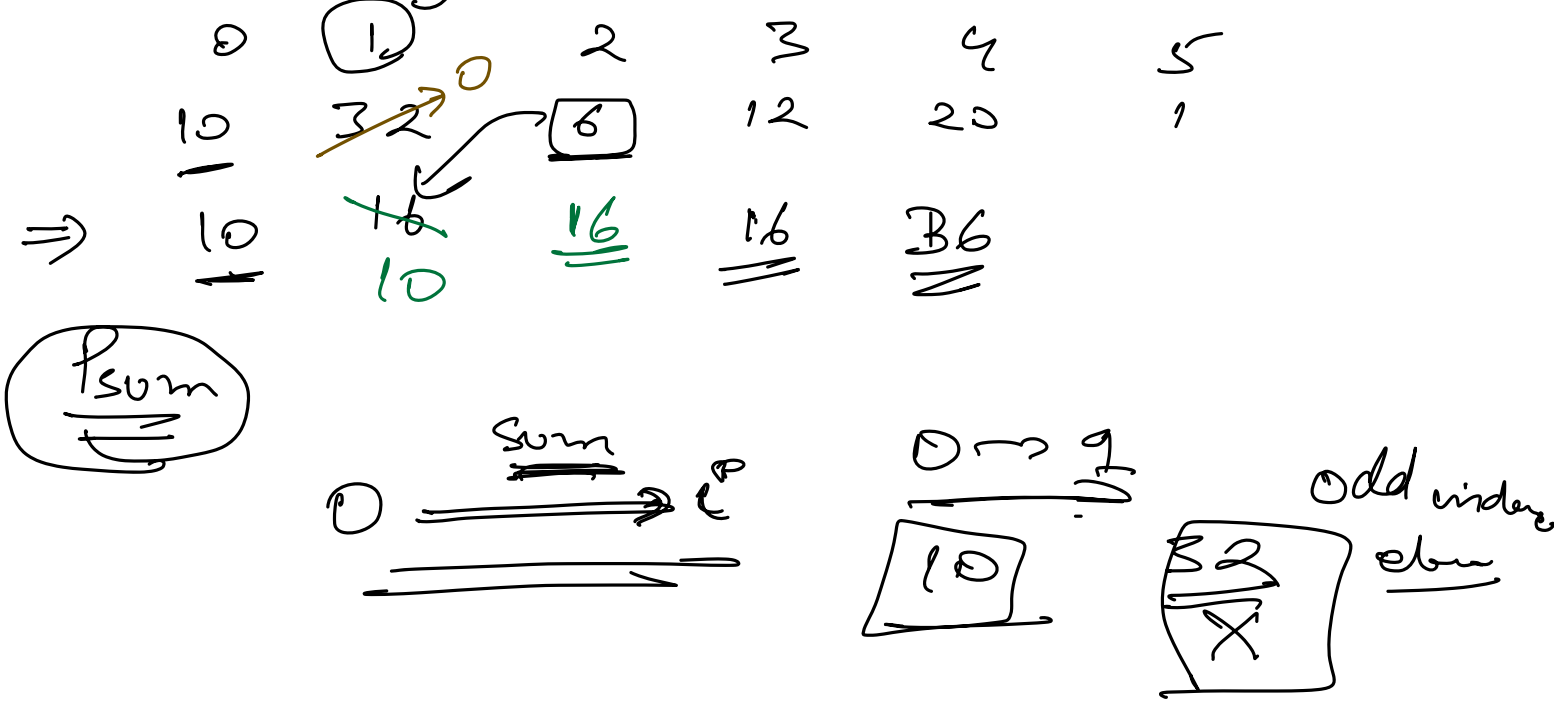
O(Q)

T.C ⇒ O(N+Q)

S.C ⇒ O(N) → O(1)

H.W ⇒ Do the same for odd indexed elements.

even indexed elements.


0 1 2 3 4 5
10 32 6 12 20 1
0
⇒ 10 16 16 86
10
Psum
sum
0 i
0 → 9
10
32
X
odd index
ebm