Prefix Sum

Building Pfsum array

Answering sum queries

Q1 : sum of even indexed

Q2 : Special Index

```
func (int N) {
    int arr[10]  → 40 B
    int x        → 4 B
    int y        → 4 B
    long z       → 8 B
    int arr[N]   → 4N B
}
```

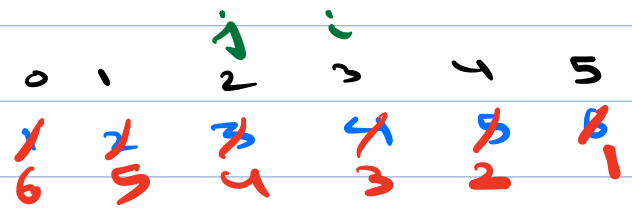
Total auxiliary space = 56 + 4N

SC = O(N)

TC of accessing arr[i] in Array arr of size N : O(1)

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

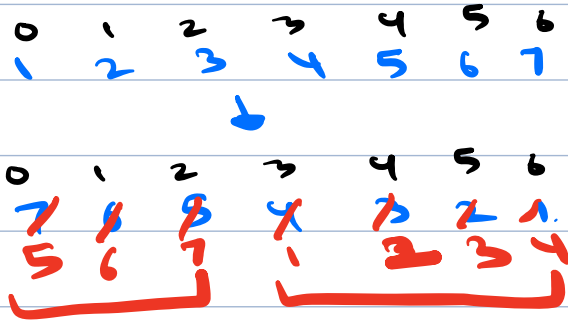
N=6




Reverse the array

```
func (arr, N, K) {
    reverse (arr, N, 0, N-1)
    reverse (arr, N, 0, K-1)
    reverse (arr, N, K, N-1)
}
```

N=7 K=3




Rotating arr of N size K times (R → L)

Q. Given stock's profit/loss for each day, develop a feature where you've to return total profit/loss between the given start and end day (both inclusive).

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

stock[] = <-5, 10, 20, 40, 50, -10, 80, -90, -20, -10>

| Start Day | End Day | Net Profit / Loss |
|---|---|---|
| 1 | 4 | 120 |
| 0 | 0 | -5 |
| 7 | 9 | -120 |
| 0 | 9 | 65 |
| 2 | 7 | 90 |

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

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

arr[] = <-3, 6, 2, 4, 5, 2, 8, -9, 3, 1>

Q = 3

| L | R | solution |
|---|---|---|
| 4 | 8 | 9 |
| 3 | 7 | 10 |
| 1 | 3 | 12 |

① Range sum
② Multiple queries

Brute Force : For each query Q, we iterate from L idx to R idx and calculate sum of elements

```
// arr, N, Q
for (int cnt = 1 ; cnt <= Q ; cnt++) {
    // take L and R as input
    int sum = 0
    for (i = L ; i <= R ; i++) {
        sum = sum + arr[i]
    }
    print (sum)
}
```

idx → [L R]
= R - L + 1

1 query → N iter

Q query → QN iter

TC : O(QN)  SC : O(1)

$1 \le N \le 10^5$

$1 \le Q \le 10^5$

$10^5 \times 10^5 = 10^{10}$ iter

TLE

Too many iterations

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|

scores[10] = 2, 8, 14, 29, 31, 49, 65, 79, 88, 97

Runs scored in 7th over = 65 − 49 = 16
[7 - 7] S[7] - S[6]

Runs scored from 6th - 10th over = 97 − 31 = 66
[6 - 10] S[10] - S[5]

Runs scored in 10th over = 97 − 88 = 9
S[10] - S[9]

Runs scored from 3rd - 6th over = 49 − 8 = 41
[3 - 6] S[6] - S[2]

Runs scored from 4th - 9th over = 88 − 14 = 74
S[9] - S[3]

[L - R] = S[R] - S[L-1]

Runs scored in range → subtraction
constant time

Magical array of scores
↓
cumulative Runs made from starting till ith over

① cumulative sum array, sum of any range in constant sum

② cumulative sum → prefix sum array

---

Prefix sum

① ar[N] → Pfsum[N]

② Pfsum[i] → sum of all elements from starting (0) till ith id

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| A[5] = | 2 | 5 | -1 | 7 | 1 |
| Pf[5] = | 2 | 7 | 6 | 13 | 14 |

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| A[6] = | 10 | 32 | 6 | 12 | 20 | 1 |
| Pf[6] = | 10 | 42 | 48 | 60 | 80 | 81 |

Q. Creating Pf array

Pf[0] = a[0]

Pf[1] = a[0] + a[1]

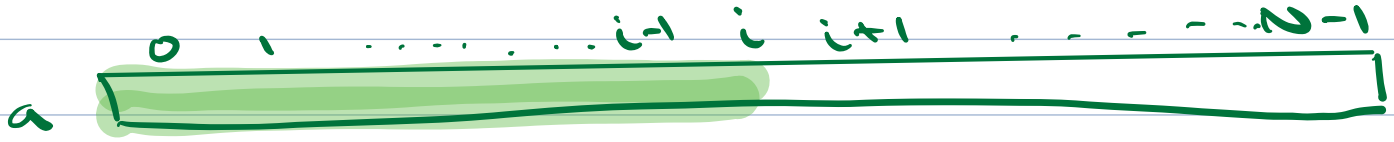
= Pf[0] + a[1]

Pf[2] = a[0] + a[1] + a[2]

Pf[2] = Pf[1] + a[2]

pf[3] = a[0] + a[1] + a[2] + a[3]

pf[3] = pf[2] + a[3]




pf[i] = pf[i-1] + a[i]

sum(0→i) sum(0→i-1)

TC: O(N)

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

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| A[5] = | [2 | 5 | -1 | 7 | 1] |
| pf[5] = | [2 | 7 | 6 | 13 | 14] |

① Sum(L-R) = pf[R] - pf[L-1]

If L==0

① sum(0-R) = pf[R]

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

**Q = 3**

| L | R | solution |
|---|---|---|
| 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 |

Q. sum(3 → 7)

pf[7] = sum(0 → 7)

= sum(0 → 2) + sum(3 → 7)

pf[7] = pf[2] + sum(3 → 7)

pf[7] - pf[2] = sum(3 → 7)

```
// arr[], N, Q

① // calculate pf[]
   int pf[N]
   pf[0] = a[0]
   for(i=1 ; i<N ; i++){
   |  pf[i] = pf[i-1] + a[i]
   }
```

N

② for ( cnt = 1 ; cnt ≤ Q ; cnt++ ) {

Q

// input L and R ; sum (L → R)

if (L == 0)

sum = pf[R]

else

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

print (sum)

}

1 query → O(1)

Q query → Q × O(1)

O(Q)

TC : O (N + Q)

SC : O (N) → pf[ ]

10:43

Given N array elements & Q queries, for each query calculate sum of all even indexed elements from L to R.

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

| Queries: | | | sum |
|---|---|---|---|
| | 1 | 3 | a[2] = 1 |
| | 2 | 5 | a[2] + a[4] = 1 + 4 = 5 |
| | 0 | 4 | a[0] + a[2] + a[4] = 2 + 1 + 4 = 7 |
| | 3 | 3 | 0 |

① Range sum ② Multiple Queries

BF : For each query, iterate from L-R, if current idx is even, add a[i] to sum.

TC : O(Q x N)

Optimized : Build pf sum

pf[i] = sum of even indexed elements from 0 → i

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

if (i % 2 == 0)

pf[i] = pf[i-1] + a[i]

else

pf[i] = pf[i-1] + 0

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| A[5] = | 2 | 4 | 3 | 1 | 5 |
| pf[5] = | 2 | 2 | 5 | 5 | 10 |

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

| Queries | L | R | |
|---|---|---|---|
| | 1 | 3 | pf[3] - pf[0] = 3 - 2 = 1 |
| | 2 | 5 | pf[5] - pf[1] = 7 - 2 = 5 |
| | 0 | 4 | pf[4] = 7 |

Sum(L-R) = pf[R] - pf[L-1]

```
// a[], N, Q
int pfc[N]
pfc[0] = a[0]
for (i=1 ; i < N ; i++) {
    if (i % 2 == 0)
        pfc[i] = pfc[i-1] + a[i]
    else
        pfc[i] = pfc[i-1] + 0
}

for ( cnt = 1 ; cnt <= Q ; cnt++) {
    // input L -> R
    if (L == 0)
        sum = pfc[R]
    else
        sum = pfc[R] - pfc[L-1]
    print (sum)
}
```

TC : O(N + Q)

SC : O(N) → pfc[]

SC : O(1) if we modify a[] to contain pf sum information

# Sum of odd indexed elements

① pfo[N]

② pfo[i] = sum of odd indexed elements from 0 to i

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

---

Google ⇒ Special Index

An index is said to be special index, if after deleting it,

sum of all even index = sum of all odd index

Count how many special index are there?

| | 0 | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|---|
| Ex | 4 | 3 | 2 | 7 | 6 | -2 | ans = 2 |

A C J

| i | 0 | 1 | 2 | 3 | 4 | $S_c$ | $S_o$ | |
|---|---|---|---|---|---|---|---|---|
| 0 | 3 | 2 | 7 | 6 | -2 | 8 | 8 | ✓ |
| 1 | 4 | 2 | 7 | 6 | -2 | 9 | 8 | ✗ |
| 2 | 4 | 3 | 7 | 6 | -2 | 9 | 9 | ✓ |
| 3 | 4 | 3 | 2 | 6 | -2 | 4 | 9 | ✗ |
| 4 | | | | | | | | ✗ |
| 5 | | | | | | | | ✗ |

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| Q : | 4 | 1 | 3 | 7 | 10 |

sum of odd indices after idx 2 is removed

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| | 4 | 1 | 7 | 10 |

11

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Q : | 2 | 3 | 1 | 4 | 0 | -1 | 2 | -2 | 10 | 8 |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| | 2 | 3 | 1 | 0 | -1 | 2 | -2 | 10 | 8 |

sum of odd indices after idx 3 is removed

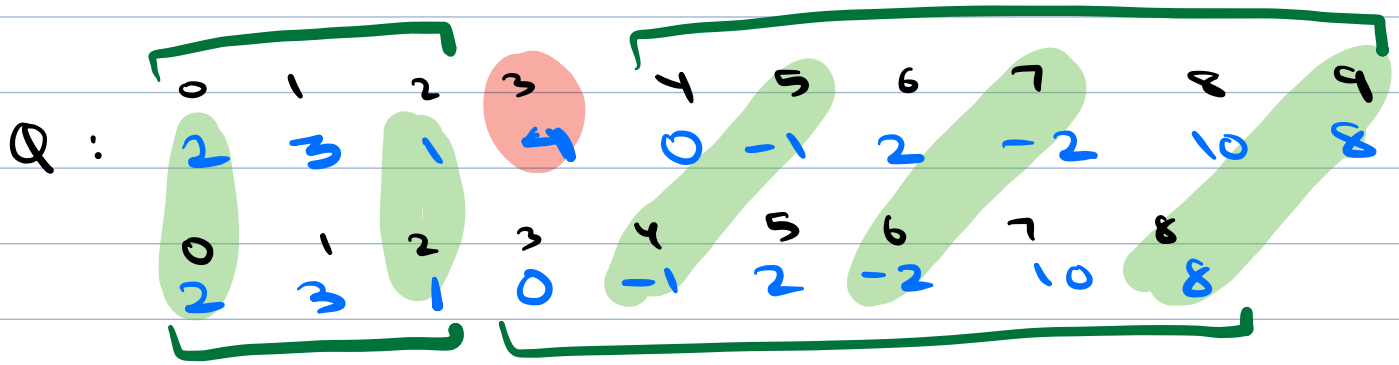
= 15

Left side + Right side

3 0 + 2 + 10

Left Right

Sum of odd (0 → 2) + sum of even (4 → last)




Sum of even indices after idx 3 is removed

Left side + Right side

Sum of even (0 → 2) + Sum of odd (4 → last)


$$sum_e = \underbrace{sum_e(0 \to i-1)}_{\text{Left } a} + \underbrace{sum_o(i+1 \to N-1)}_{\text{Right } b}$$

$$sum_o = sum_o(0 \to i-1) + sum_e(i+1 \to N-1)$$

$x$ + $y$

```
// Create pfo[N] and pfe[N]
int cnt=0

if (i==0) {
    sum_odd_in_new = pfe[N-1] - pfe[0]
    sum_even_in_new = pfo[N-1] - pfo[0]
    if (sum_odd_in_new == sum_even_in_new) {
        cnt++
    }
}

for (i=1 ; i < N ; i++) {
    x = pfo[i-1]
    y = pfe[N-1] - pfe[i]
    sum_odd_in_new = x + y          i+1 → N-1

    a = pfe[i-1]
    b = pfo[N-1] - pfo[i]
    sum_even_in_new = a + b

    if (sum_odd_in_new == sum_even_in_new)
        cnt++
}

return cnt
```

3N

TC: O(N)

SC: O(N) = 2N

↓

Hp and Hc

---

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| A = | 1 | 2 | 3 | 4 |

B = 7

(i, j)

i != j