Problem Solving 1

1) Prefix Sum → [Range] ≤ 100
2) Pick from Both Sides
3) Query prefix Sum
4) Peak from Left & Right
5) Rain Water Trapped
6) Missing Integer

Array[] : 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

Pf[i] = sum of all elements from [0, i]

Pf[i], using Pf[i-1]

$\sum_{i=0}^{N-1}$ Pf[i] = Pf[i-1] + arr[i]

[0, i] = [0, i-1] + arr[i]

base work i=0 | Array size & bounds

Pf[0] = arr[0]

$\sum_{i=1}^{N-1}$ Pf[i] = Pf[i-1] + arr[i]

Iterations: (N-1), TC: O(N)

SC: O(N)

Query:

[l, r] : Pf[r] - Pf[l-1] + arr[l]

[0 - r] - [0 - l-1]

[l, r] = arr[r] - [l, r]

Pick from Both Sides [2 ptr] [Array]

Given an array arr[N] we can pick k elements from either the left or right end of the array. Find the return max possible sum

Notes: At each step we can pick an element an element either from

Pick end or left end after picking k from is detected

Notes: k > n

Array

Ex: arr[] = {3, -2, 1, 3, 3}

B=3

ans = 8

Ex: arr[5] = {2, 3, 10, 2, 3}

B=4

Sum = 1 left = 2, Right = 2, -1, 6

Sum = 15, left = 2, -3, 10 Right = 1

Sol: Take Max k elements

Sol: Array of pick Max k elements X

Ex: arr[5] : -2 1 100 2 6

B=2 : sum = 8

-2 2 1 6 100

Sol2: At every step pick up max & remove elements, longest window work.

Sol3: Take first k elements sum vs Last k elements sum X

Ex2: arr[10] = {3, -4, 6, -5, 2, 3, 2, -1, 8, -2}

B=5

Left : [0, 5] = 2,

left : [0, 2] = 0

S = S - Ar[4]

left = [0, 2] =

S = S - Ar[2]

= 0 - (-5) = 5

left = [0, 1]

S = S - Ar[2]

= -1

Right = 0

Right = -2

[4 - 5] = -2

Right = 1 = 1 + Ar[2]

[2 - 4] = 1 - 6

Right

[7 - 4] = 1 = 1 + Ar[2]

1 = 1 - 5

TC: O(k) → O(N)

SC: O(1)

Bruteforce:

for L, R = 0, ans = 0

T = 0; for k, B, T++ {

L = L + ar[T]

}

ans = L + R }

L: 0, 1, 2, ... B-1

L: 0, 1, 2, ... B-2

L: 0, 1, 2, ... B-3 (0 → 2)

R: 0

R: N-1

R: N-2, N-1

J = N-1;

T = B-1; T >= 0; T--{

L = L - ar[T]

R = R + ar[J]

J--;

ans = max (ans, L+R)

}

return ans

Given an arr[N], construct left[N], right[N]

left[i] : Should contain max of all elements from [0, i]

Right[i] : Should contain max of all element from [i, N-1]

arr[]: 0 1 2 3 4 5 6 7
2 6 4 2 5 1 3 2 5

left[]: 2 6 6 6 8 8 8 8 8

Right[]: 8 8 8 8 8 3 3 2 0

left[0] = max of all [0, 0]

left[i-1], arr[i]

left[i] = max(left[i-1], arr[i])

left[0] = arr[0]

for i, i < N; i++{

left[i] = max (left[i-1], arr[i])

}

TC: O(N)

Right[N-1] = arr[N-1]

Right[N-1], arr[N-1]

Right[i] = max(arr[i], Right[i+1])

for i = N-2; i >= 0; i--{

Right[i] = max (arr[i], Right[i+1])

}

TC: O(N)

5 6 7 8

Right[i] = max [5 - 8]

Right[6] = max [6 - 8]

Right[8] = max (arr[8], Right[8])

Given arr[N], Each arr[i] represents height of i th building of each building

Water = ?, On a rainy day calculate how much water is accumulated?

Ex: arr[5] = {4, 3, 5, 1, 4}

Ex: arr[15] = {4, 1, 7, 5, 3, 1, 7, 5, 3, 7, 3, 5, 1, 4, 2, 7, 3, 5, 1}

Idea = Σ (water accumulated on top of each building)

for a block? we consider left & Right

left[i], Right[i]

How much water is present on top of i th building

$\sum_{i=0}^{N-1}$ min (left[i], Right[i]) - arr[i]

Summation

left[i] = max [0, i]

> = arr[i]

Right[i] = max [i, N-1]

>= arr[i]

min (left[i], Right[i]) >= arr[i]

11:03PM

Ex: arr[N] = 0

Q: 2

L R Q

for every query, add 1 to all index from L, R

1 - 2 5 2
2 - 4 7 2
3 - 1 11 1
5 - 6 13 2
7 - 7 -3 1

print final arr[]

arr[10] = 0 0 0 0 0 0 0 0 0 0

final arr[10]:

0 -4 -1 1 1 6 0 0 -2 -3

Sol:

For every Q:

for L, i, R = Q[i, 1, 2]{

arr[L] += x

}

TC: Q × N

SC: O(1)

arr[10] = 0 0 0 0 0 0 0 0 0 0

-4 3 2 -3 -2 -1

arr[N] = 0 -4 3 2 0 0 -3 -2 -1

Pf[] = 0 -4 -1 1 1 6 0 0 -2 -3

Q: - 2 5 2 → arr[2] += 2, arr[5] -= 2

3 - 4 7 2 → arr[3] += 2, arr[7] -= 2

1 - 1 11 1 → arr[1] += (-1), arr[11] -= (-1)

5 - 6 13 2 → arr[5] += 2, arr[13] -= 2

7 - 7 -3 1 → arr[7] += -3, arr[-3] -= 1

For every Q:

L, R, x : arr[L] += x,

arr[R+1] -= x {if R+1 > (N-1), on then update}

TC: O(Q)

→ Calculate prefix sum for array → TC: O(N)

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

Given an arr[N], it contains all the elements from 1 - N+1, except 1 element, find the missing element.

Ex1: arr[5] = {1, 3, 2, 5, 1, 4}

Ex2: arr[7] = {7, 1, 3, 2, 5, 1, 3}

Sol:

1) Brute force check: N log N

Sum of All Ele 1 - (N+1) − Sum of All array elements

Sum of all Elements from 1 - (N+1)

Sum of all array elements

Sum of all Elements from 1 - (N+1)

Sum of all array elements

Sum of all Elements from 1 - (N+1)

Sum of all