Satya Sai Siva Rama Krishna → 3 years : Telugu

Advanced Content : Pseudocodes language independent

↳ Convert in your language of choice

Todays Content:

10:23 —10m→ 10:33pm

a) Zero Queries
b) Water logging
c) Max Subarray sum
d) Flip, if time permits

IQ) Given a ar[N], initially ar[] = 0, return the final array after performing all the queries, {All Q Queries}

Queries (i, n) = Add n to all the numbers from A[i] to A[N-1]

ar[7] = {0}

Ex: A = [0 0 0 0 0 0 0] (i, n) : 3 Queries

0	1	2	3	4	5	6	
3	3	3	3	3	3		(1 3)
				-2	-2	-2	(4 -2)
			4	2	2	2	(3 1)

A = [0 3 3 4 2 2 2]

ar[7] = {0}

Ex: A = [0 0 0 0 0 0 0] [i, n] → Queries

0	1	2	3	4	5	6		
		6	6	6	6	6	2	6
-1	-1	5	5	5	5	5	0	-1
			7	7	7	7	3	2
					11	11	5	4

A = [-1 -1 5 7 7 11 11]

Ideal: For every query [i, n] iterate from ar[i] → ar[N-1] & add n

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

Idea:

Ex: A = [0 0 0 0 0 0 0]

0 1 2 3 4 5 6

0 3 0 1 -2 0 0

Apply Pf = 0 3 3 4 2 2 2

(i, x) : 3 Queries

(1 3)

(4 -2)

(3 1)

Ex: A = [0 0 0 0 0 0 0]

0 1 2 3 4 5 6

-1 0 2 2 0 0 0

Apply Pf = -1 -1 1 3 3 3 3

[i, x] → Queries

2 6

0 -1

3 2

2 -4

Cross check:

A = [0 0 0 0 0 0 0]

0 1 2 3 4 5 6

i	x	
2	6	
0	-1	-1 -1
3	2	
2	-4	1 3 3 3 3

A = [-1 -1 1 3 3 3 3]

```
int[] modify(int ar[], int N, int Q, int mat[Q][2]){
```

↳ for a^{th} Query: $i = mat[a][0]$, $x = mat[a][1]$

Step 1: Iterate m queries update array

```
for(l = 0; l < Q; l++){
    i = mat[l][0]   x = mat[l][1]
    ar[i] += x
}
```

Step-2: Apply pfsum on modified ar[]

```
}
```

TC: $Q * O(1) + O(N) \rightarrow O(N+Q)$

SC: prefix in ar[] → SC: O(1)] TODO

→ // modifying ar[] directly

prefix wa pf[] → SC: O(N)

→ // Taking extra extra to apply pf[]

ar[N] = 0 { Perform all Q Queries & return final array }

Queries → (i, j, x) = Add x to all numbers from A[i] → A[j]

	0	1	2	3	4	5	6			
A[] =	0	~~0~~	~~0~~	~~0~~	0	0	0	i	j	x
		2	2	2				[1	3	2]
			~~3~~	~~3~~	~~3~~	3	→	[2	5	3]
				4	4	2		[2	4	-1]

A[] = [0 2 4 4 2 3 0]

Ideas: For every query [i, j x], add x for all elements [i-j]

↳ TC: Q * N SC: O(1) → // modifying ar[] = 0

Hint:

(i, j, x):

0 1 2 .. i i+1 ... j j+1 j+2 .... N-1

[Add x i, N-1] = x x ... x x x x

[Add -x j+1, N-1] -x -x .... -x

(i, j x) = [Add x from i, N-1] [Add -x j+1 N-1]

= Query (i, x) Query (j+1, -x)

= ar[i] += x ar[j+1] -= x

A[] = [0 ∅ ∅ 0 ∅ ∅ ∅]

2 ∅ -2 -3

2 1

A[] = 0 2 2 0 -2 1 -3

Apply pf[] = 0 2 4 4 2 3 0

i j x → ar[i] += x ; ar[j+1] -= x

[1 3 2]

[2 5 3]

[2 4 -1]

A[] = [0 ∅ ∅ ∅ ∅ ∅ 0]

4 6 2 -4 -6

A[] = [0 4 6 2 -4 -6 0]

pref = [0 4 10 12 8 2 2]

i j x → ar[i] += x ; ar[j+1] -= x

2 4 6

3 6 2 : ar[7] : out of bounds no need to subtract

1 3 4

```
int[] modify(int ar[], int N, int Q, int mat[Q][3]){
    Qury : [i j x]
    Step1: update all queries
    for(l=0; l<Q; l++){
        i = mat[l][0], j = mat[l][1], x = mat[l][2]
        ar[i] += x
        if(j+1 < N){
            ar[j+1] -= x
        }
    }
    Step2: Apply pf[] on modified ar[]
}
```

TC: O(Q + N) ⇒ O(N+Q) SC: O(1), O(N) → O(N) if we take extra ar[]; O(1) if we modify given ar[]

2Q) Given ar[N],

Construct pfM[N] s.t, pfM[i] = max of all elements [0-i]

	0	1	2	3	4	5
ar[6] = {	1	-6	3	2	8	7 }
pfM[6] = {	1	1	3	3	8	8 }

	0	1	2	3	4
ar[5] = {	3	-2	6	2	8 }
pfM[5] = {	3	3	6	6	8 }

TODO: Construct pfM[] → TC: O(N)

3Q) Given ar[N],

Construct sfM[N] s.t, sfM[i] = max of all elements [i, N-1]

	0	1	2	3	4	5	6
ar[7] = {	3	10	6	7	0	2	-1 }
sfm[7] = {	10	10	7	7	2	2	-1 }

	0	1	2	3	4
ar[5] = {	4	8	-1	6	3 }
sfm[5] = {	8	8	6	6	3 }

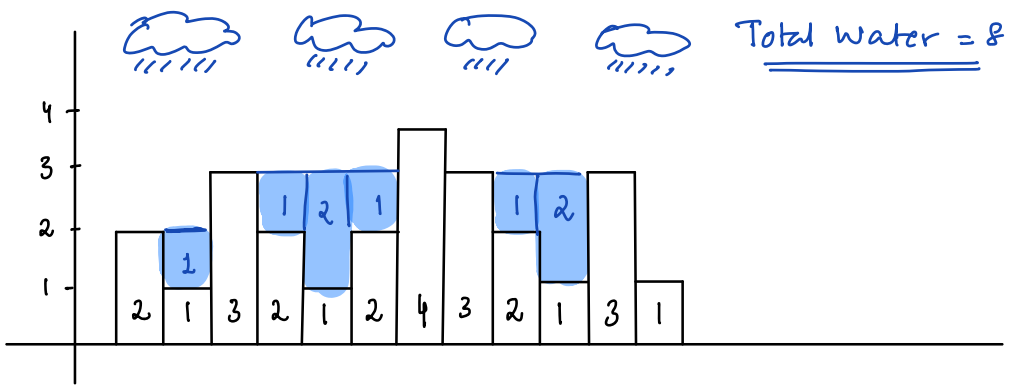
TODO: Construct spM[] → TC: O(N)

48) Rain water trapped ?

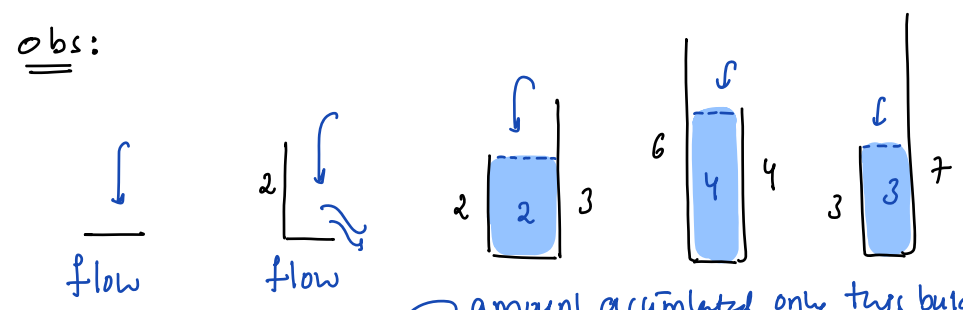
Given ar[N] elements, where ar[i] represents height of the building, return amount of water trapped in all buildings

Note: Width of each building is 1

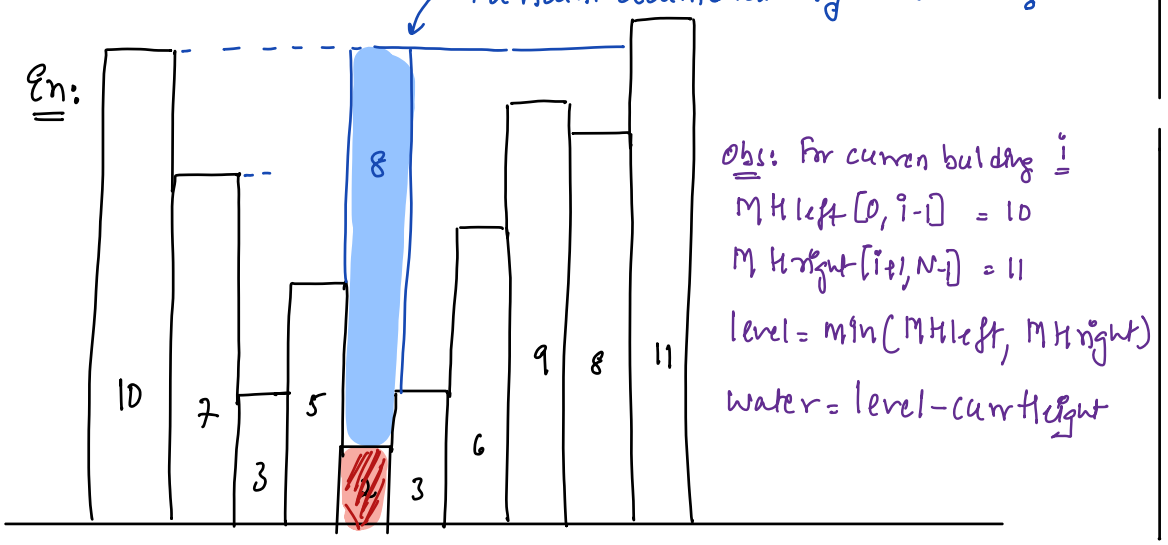
	0	1	2	3	4	5	6	7	8	9	10	11
ar[ ] = {	2	1	3	2	1	2	4	3	2	1	3	1 }



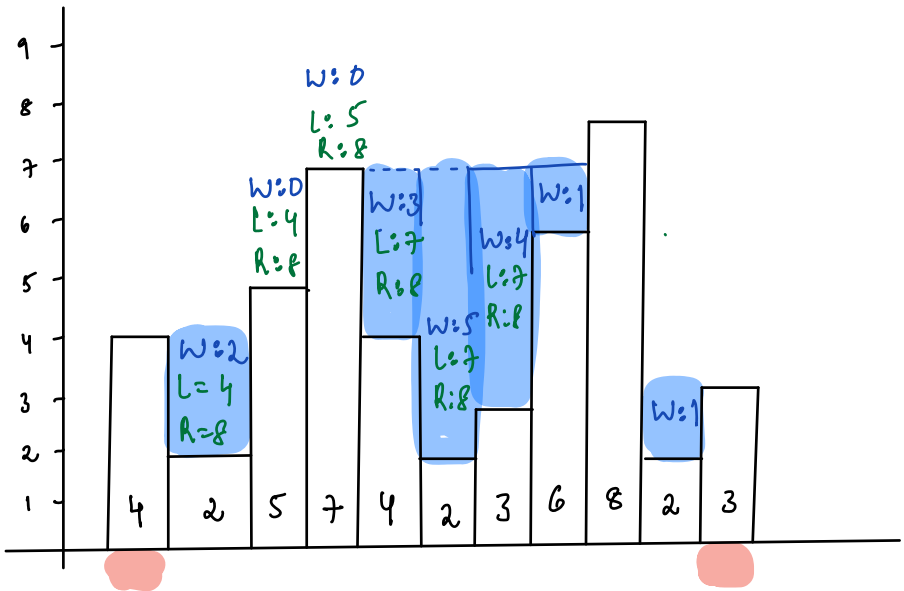
obs:



Eg:



Ex:



Lman :	4	4	5	7	7	7	7	7	8	
Rman :	8	8	8	8	8	8	8	3	3	
Level :	4	4	5	7	7	7	7	3	3	
Water :	2	-1	-2	3	5	4	1	-5	1	= water = 16

if -ve consider : 0

obs: skip corners, because water cannot be accumlated

```
int Water(int H[], int N){   TC: O(N+N+N) ⇒ O(N)
                             SC: O(N+N) ⇒ O(N)
    int ans = 0
    int pfM[n] // Construct → O(N)
    int sfM[n] // Construct → O(N)
                                  ↳ no water in corners
    i=1; i<n-1; i++){ // We neglect i=0 & i=n-1 → O(N)
        // for ith building
        Lman = max of ar[] from [0, i-1] = pfM[i-1]
        Rman = max of ar[] from [i+1, N-1] = sfM[i+1]
        level = min(Lman, Rman)
        water = level - H[i] // level water can rise - current
                                                  height of building
        if(water > 0) {ans = ans + water}
    }
    return ans;
```

All my recordings in What's app

1) Slack / What's-app
2) Interviews
   a) Advanced DSA
   b) HW / CW
   c) Previously asked interview question of that company / leetcode /
3) SDE-1 ⇒ DSA
4) SDE-2 ⇒ DSA + Design (LLD)

SDE-3 : less DSA + More Design (LLD & HLD)

4) Duration : 55 Sessions : 4½