#Segment Tree#

Concepts & Qns...

Facebook
Instagram → codestorywithMIK

(Twitter) → CSwithMIK

codestorywithMIK →

"No more fear of Segment Tree"

#Motivation

" So many success stories. Next one is going to be yours "

video - 5

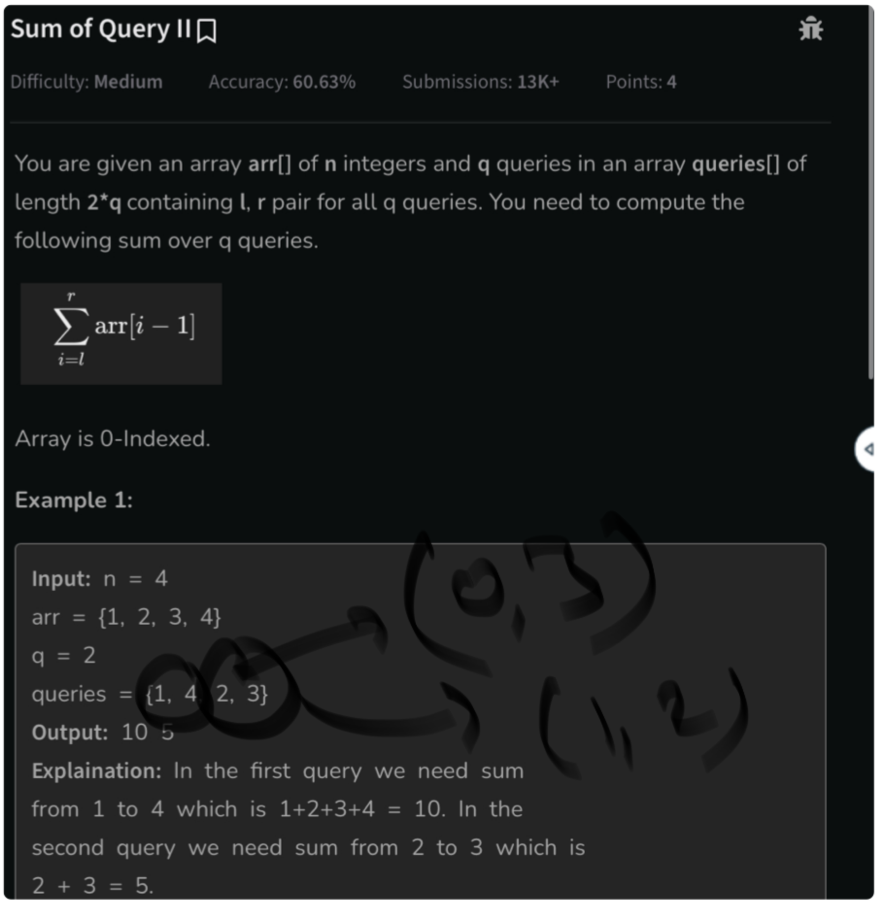
Recap :-

- ✓ we understood about segment Tree ? what ? why ? when ?
- ✓ buildSegmentTree
- ✓ Example - Range Sum in an array
- ✓ Update Query
- ✓ Range Query
- ✓ why take 4*n size array

Sum Of Query - II

GFG

amazon



Sum of Query II

Difficulty: **Medium** Accuracy: **60.63%** Submissions: **13K+** Points: **4**

You are given an array **arr[]** of **n** integers and **q** queries in an array **queries[]** of length **2*q** containing **l**, **r** pair for all q queries. You need to compute the following sum over q queries.

$$\sum_{i=l}^{r} \mathrm{arr}[i-1]$$

Array is 0-Indexed.

Example 1:

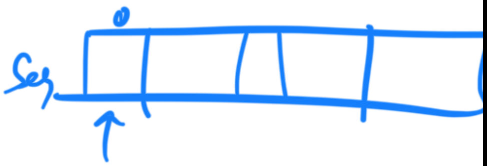
```
Input: n = 4
arr = {1, 2, 3, 4}
q = 2
queries = {1, 4, 2, 3}
Output: 10 5
Explaination: In the first query we need sum
from 1 to 4 which is 1+2+3+4 = 10. In the
second query we need sum from 2 to 3 which is
2 + 3 = 5.
```

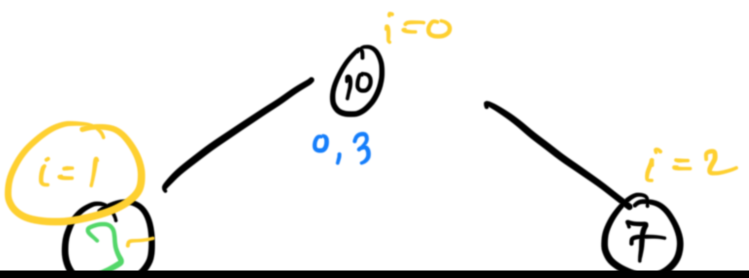
~~Seg~~

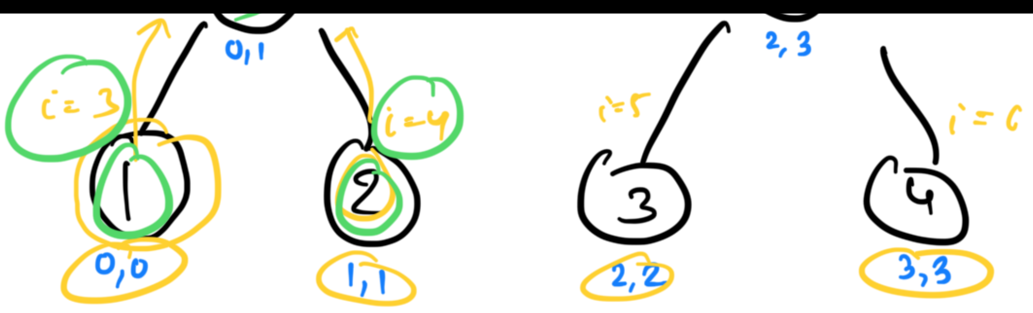
Steps:-

① Build SegmentTree (0, n-1, segmentTree, 0);

$$arr = \{\overset{0}{\underset{l}{1}}, \overset{1}{2}, \overset{2}{3}, \overset{3}{\underset{r}{4}}\}$$







```
void - Builds ( int l,  int r,  segTree,  i ) {
        if (l == r) {
             segTree [i] = arr[l]; return;
        }

        int   mid = l+(r-l)/2;
        Build (l, mid, segTree, 2*i+1);
        Build (mid+1, r, segTree, 2*i+2);

        segment[i] = segTree[2*i+1] + segTr[2*i+2];
}
```

query → (start, end) Sum.

query (start, end, 0, 0, n-1, segTree);

```
int quey (start, end, i, l, r, segTree) {
        if(l > end || r < start) {
              return 0;
        }

        if (l >= start && r<=end) {
              return segTree[i];
        }

        mid = l+(r-l)/2;
        return quey(start, end, 2*i+1, l, mid, sTree)
           + qu (st, end, 2*i+2, mid+1, r, seg);
}
```