Lecture :- Contest 2

Agenda

- Complicated numbers
- Unset bits in a range
- Building height.

Qu1 **complicated numbers**

Given arr[n]. Return the complicated numbers in the array. in same **relative ordering** as original array.

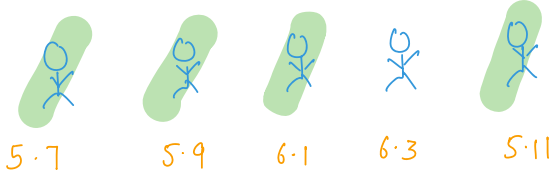
complicated numbers: el which have **at least 2 numbers greater than themselves**

Ex: arr[] = [2 3 1 4 3] ⇒ ans = [2 , 1]

2 → 3, 4, 3
3 → 4
1 → 2, 3, 4, 3
3 → 4

constraints: $1 <= n <= 10^5$

$O(n^2)$ ⟶ TLE



5·7 5·9 6·1 6·3 5·11

max = 6·3

All people having at least one person greater than him/her self.

[2 , 3 , 1 , 4 , 3]

All el having at least 2 el greater than itself.

smax = 3.

— i = 0 — i = n-1

i = 0 [2] — 2 < 3 complicated

2 less than smax

2 less than max.

i = 1 [3] → 3 not less than smax

↓

never be complicated

Logic — find smax in the array.

Iterate i from 0 to n-1 —

if arr[i] < smax — complicated

arr[] = [6, 6, 6, 3] — max = 6, smax = 6.

smax = 3

arr[i] not less than 3 [not complicated]

arr[i] < 6 → complicated

solved

arr[] = [2, 2, 2] — max = 2, smax = 2.

Qu2

Given three numbers A, B and C. You would unset the C bits of A from the right. Your friend would then restore the B bits from right to their initial configuration. Return the resultant integer.

1 → 0

a = 7

b = 1

c = 3

7 = 0 0 1 1 1

1. Unset C (3) bits from right

0 0 1 1 1 = 0

0 0 0

2. Restore B (1) bit from right

0 0 0 0 1

0 0 0 0 1 = 1 Ans

7 → 1 1 1

unset 3 bits from right

0 0 0

0 0 0

restore 2 bits from right

1 1

0 1 1

= 3

7 — 0 1 1

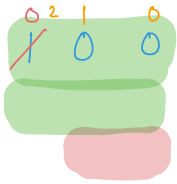
Unset 3rd bit from the right

A = 12

B = 2

C = 3

12 → 0 0 1 1 0 0 = 8




```
int solve(int A, int B, int C) {
    for(int i = B+1; i <= C; i++) {
        unsetBit(A, i-1)
    }
```

A = 12

B = 2

C = 3

→ 1. Unset 0th bit to 2 bit

2. Restore 0th bit & 1st bit

} Combine = unset 2nd bit

Given arr[n] representing height of the buildings.

find the product of heights of all buildings that are shorter than the building immediately to their right.

since, product of these heights can be very large,

return product % $10^9 + 7$.

arr[] = [2 3 1 6 9]

2 < 3 3 > 1 1 < 6 6 < 9

product = 2 * 1 * 6 = 12.

arr[] = [2 5 3 6]

2 < 5 5 > 3 3 < 6

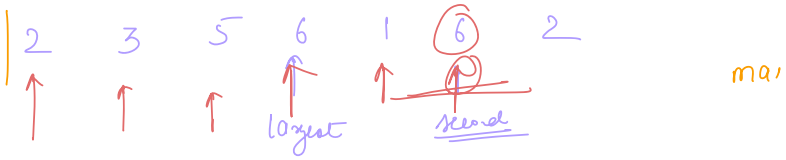
product = 2 * 3 * = 6

arr[] = [1 1 1 1 1]

product = 0

arr[i] < arr[i+1] → contribute to product

Thankyou :)



max = ~~1 2 3 5~~ 6

smax = ~~1 2 3~~ 5 ~~6~~

6 not gr to m.

```
if(arr[i] > max){
    smax = max
    max = arr[i]
} else if(arr[i] > smax)
```

}