Thursday, 25 July 2024
8:46PM

Binary Search

↳ Reduce the search space → Sorted

⇓

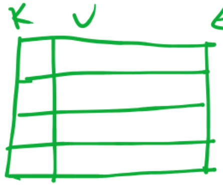
O(logn)

↳ Binary Search

Minimize the maximum

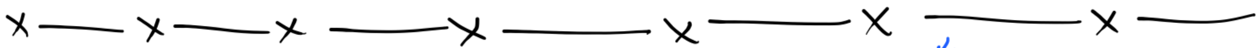
Maximize the minimum

✓Hashing → Searching in O(1) time



HashMap ✓

Searching in Set



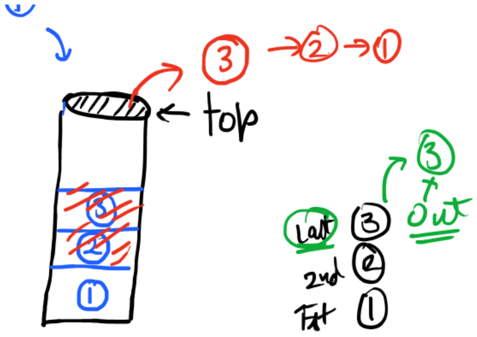
Stack

Stack of plate

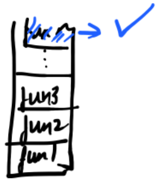
↳ ?? LIFO ✓

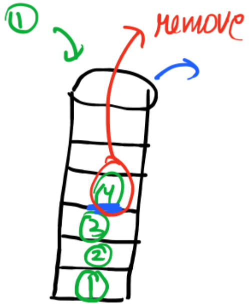
↳ Last In First Out

Stack of cards



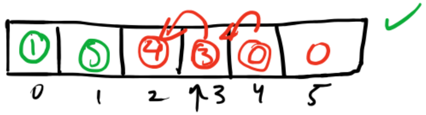
→ Stack Memory





Stack

Array / Array List



* Stacks do not have index

	Stack	Array
Adding	Only from 1 end O(1)	Directly at index O(1)
Getting If top O(1) else O(n)	Only get the top element	Directly at the index
Removing O(1) if top O(n) else	Only the top most can be removed	Shifting the other elements

O(n) → When idx is unknown
O(1) → When idx is known

O(n) → Shifting
O(1) → If end is removed

* Stack is like an array without indexes and only 1 end is available.

→ Dynamic

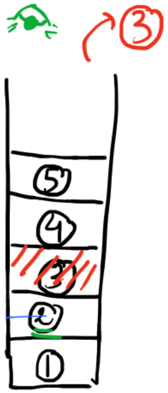
Methods of Stack

Push → Adding an element in the stack O(1)

Pop → Removes and Returns the top element O(1)

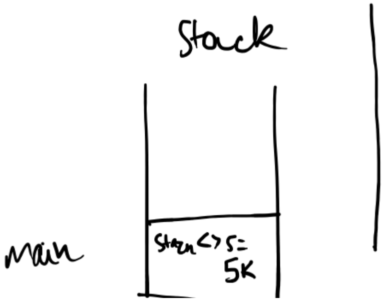
Peek → Returns the top element but doesn't remove O(1)

Size → Returns the size of the stack O(1) * with a variable



Push 1
Push 2
Push 3
Pop ⇒ 3
Peek ⇒ 2
Push 4
Peek ⇒ 4
Push 5
size → 4

* We cannot traverse a stack → Cannot use indexes.



Heap → Stack object will be present in heap memory

int vs Integer

int → ✗ objects, data type

Integer → Class → objects

Internally, Stack uses arrays.

Class on Sat → Stack ✓
11-3, 12-4 ✓

6 → contest on 100% ✓
1.5 BS + Hashing

Sunday 12-3 3.30 ✓
→ Doubt + question
BS + Hashing

()(()) (()) ✓

))((✗

)()(✗

((() ✗

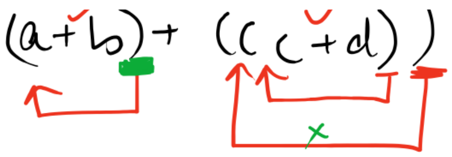
Q1 Extra Brackets ✓

$(a+b) + ((c+d))$ ✓

$((a+b)+(c+d))$

$(a+b) + ((c+d)+e)$

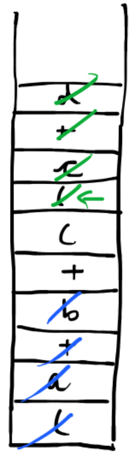
No extr
bracke



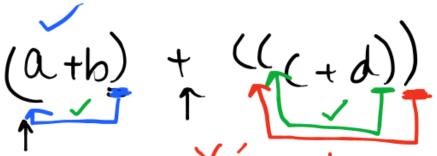
↳ we are not doing any calculation

T(c) : O(n

S(c) : O(n



⇐ top of stack



'('

↳ × Invalid

⇒ return true

⇒ extra brack.

Next Greater Right

6 5 8 0 2 3 1

8 8 -1 2 3 -1 -1

1) Using 2 loops find the greater element on right side.

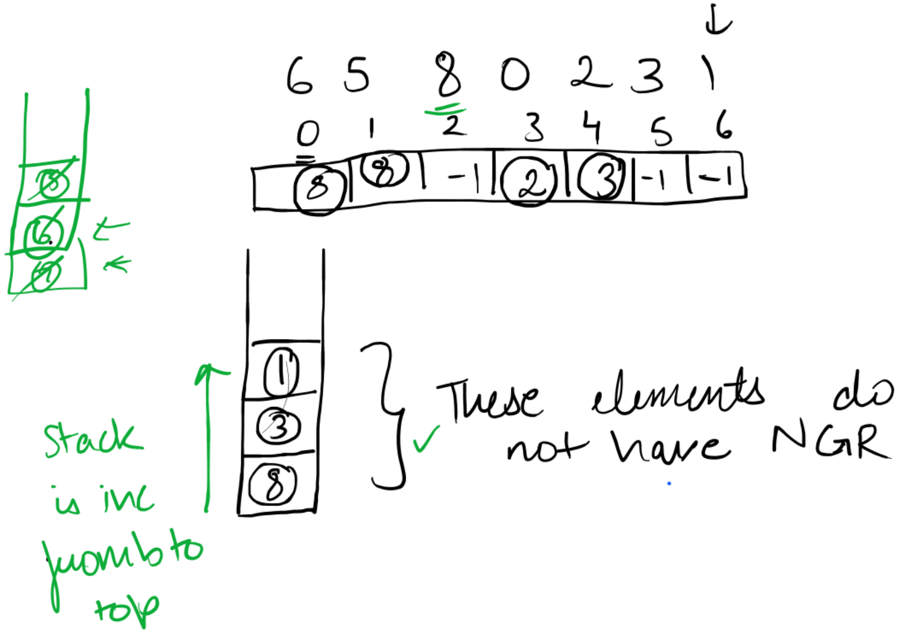
for (i=0 → n)

T... O(n²)

TC : O(n²)
SC : O(1)

```
for(j=i+1 -> n){
        if (a[j] > a[i]){
            ans[i] = a[j]
            break
        }
    }
}
```

2) Using Stack



Stack is inc from bottom to top

These elements do not have NGR ✓

Traverse from left to right

↳ while the current element is greater than the top of stac

↳ current el is the NG from to

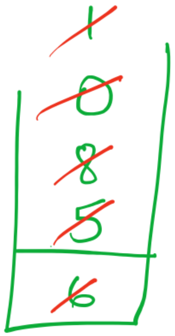
TC : O(n)
SC : O(n)

(-1) (-1) (-1) (-1) (-1)

$\underline{5}, \underline{\underline{4}}, \underline{\underline{\underline{3}}}, \underline{\underline{\underline{\underline{2}}}}, \underline{\underline{\underline{\underline{1}}}}$

0 1 2 3 $\underline{\underline{4}}$

$$n = 1 + 2 + 3 + 4 + \dots (n-1) = \frac{(n-1)(n-1+1)}{2} = \frac{n(n-}{2}$$



$\underline{\underline{6}}, \underline{\underline{5}}, \underline{\underline{8}}, \underline{\underline{0}}, \underline{\underline{1}}$

$$n = n \times = \quad = 2 \times n \approx \underline{\underline{O(n)}}$$

Created with OneNote.