Date: .....

Expt. No.: .....

Page No.: .....

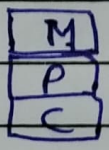
Stack's

implicit

-> In memory we do this in recursion
Memory / computer

Explicit

Real Life Examples

1) Books → 

2) Coins → 

3) Stones → 

Operations

1) Push → add O(1)

2) Pop → delete O(1)

3) Peek O(1)

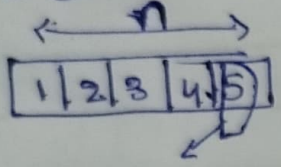
Note: Last in first out (LIFO)

Teacher's Signature _____

Implementation

Arrays

Size → fixed



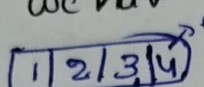
last index (top)

Stack == full

ArrayList ✓

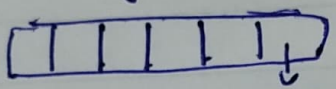
Variable → Size

internal we have array →



TOP

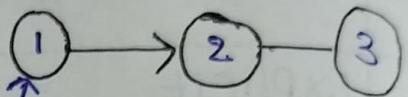
copy ←



TOP

LinkedList ✓

Variable → Size



head

O(1) → Change → Add Pop

Stack_using ArrayList

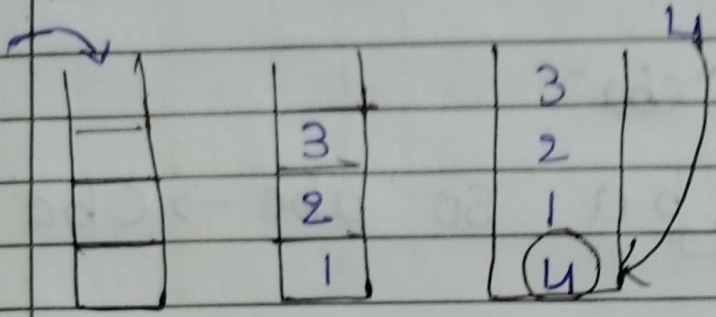
Date: .....

Expt. No.: .....

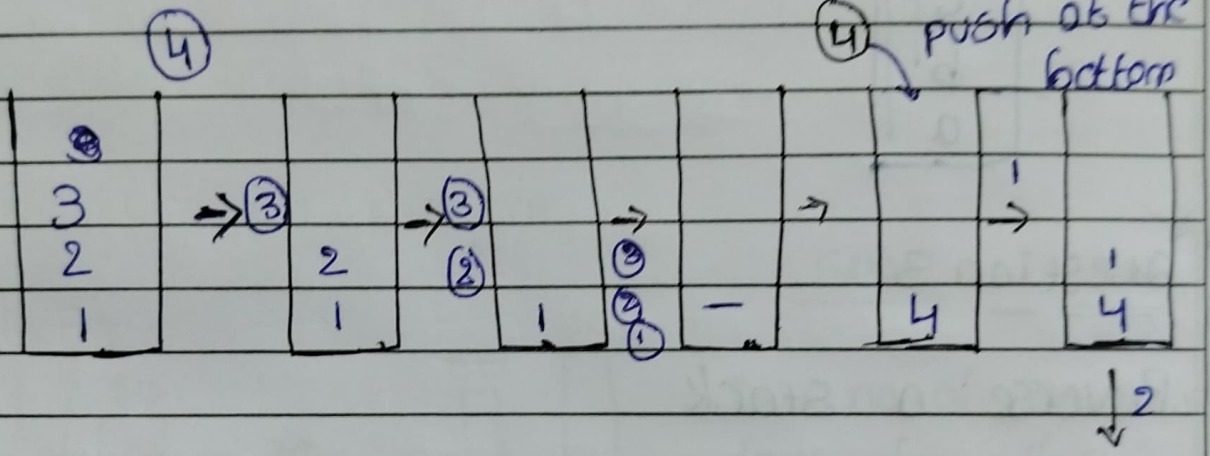
Page No.: .....

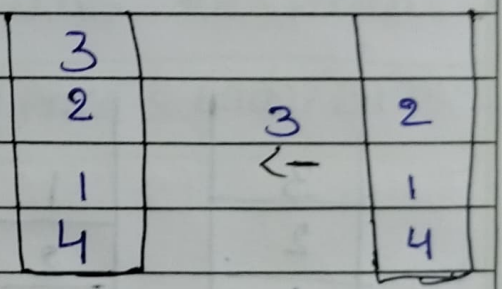
Question:

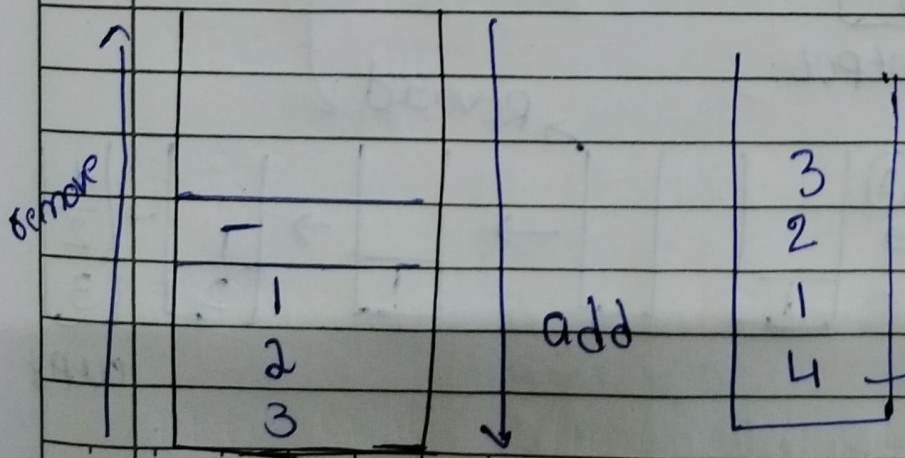
Push at the Bottom of the stack



O(1)







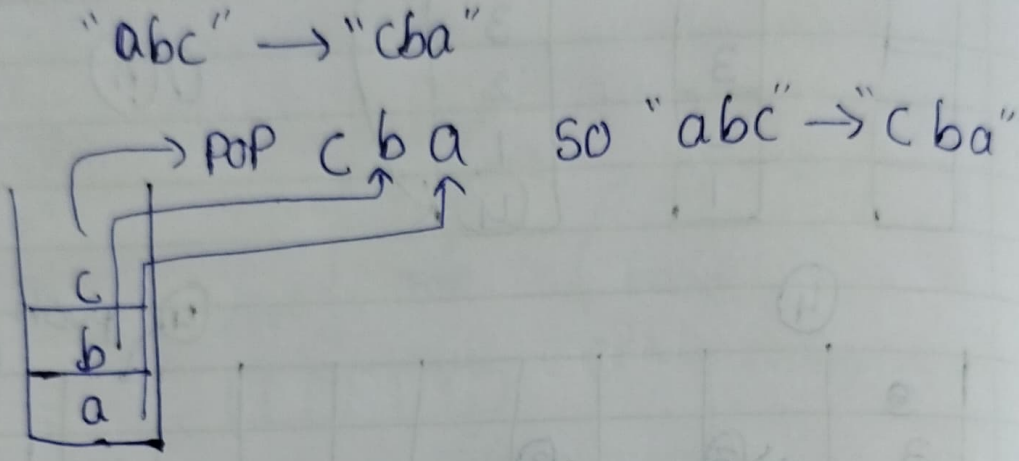
using recursion implicit

Teacher's Signature _____

Question 2

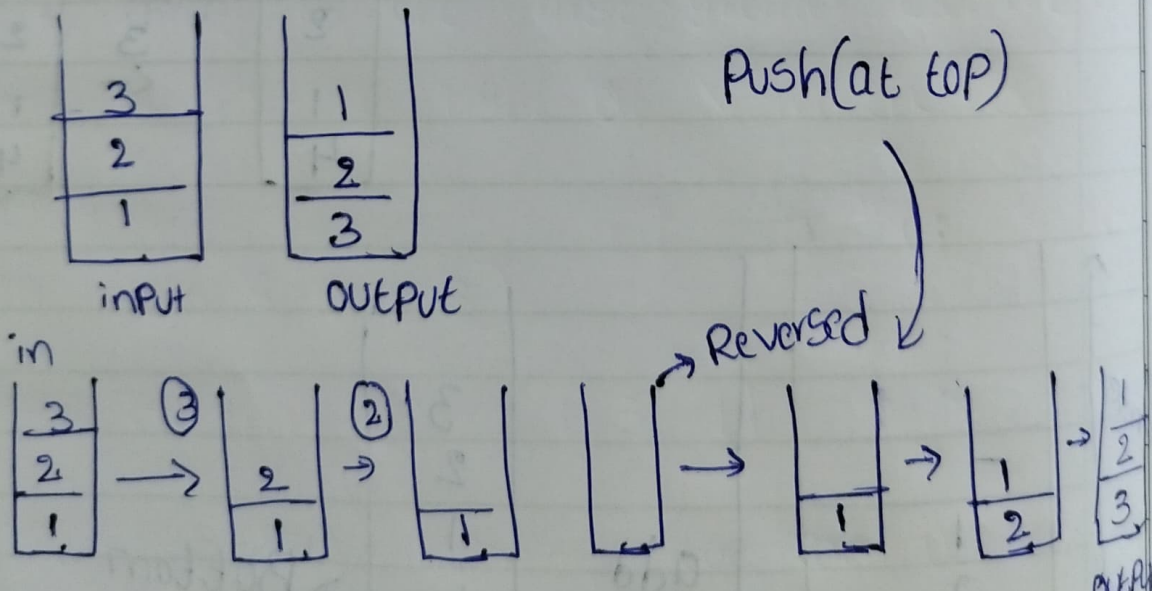
Reverse a string using a stack

"abc" → "cba"



Question 3

Reverse a stack

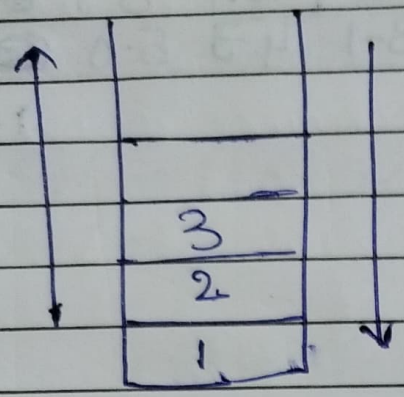


Date: .....

Expt. No.: .....

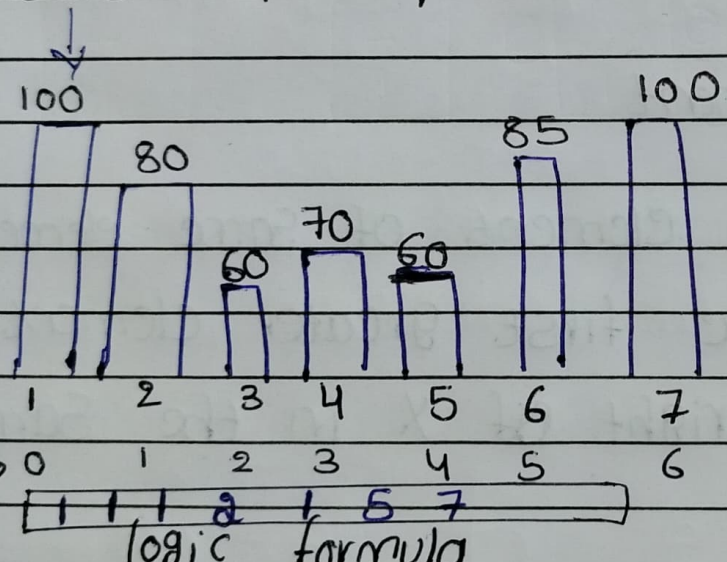
Page No.: .....

Recurssion



Question 4

Stock Span problem



Span

Max no. of Consecutive days for which

Price ≤ today's Price

logic formula

prevhigh - idx

Span = i - prevhigh

0 - 0 = 0

1 - 0 = 1

2 - 1 = 1

3 - 1 = 2

4 - 1 = 3

5 -

0 - 0 = 0

1 - 0 = 1

2 - 1 = 1

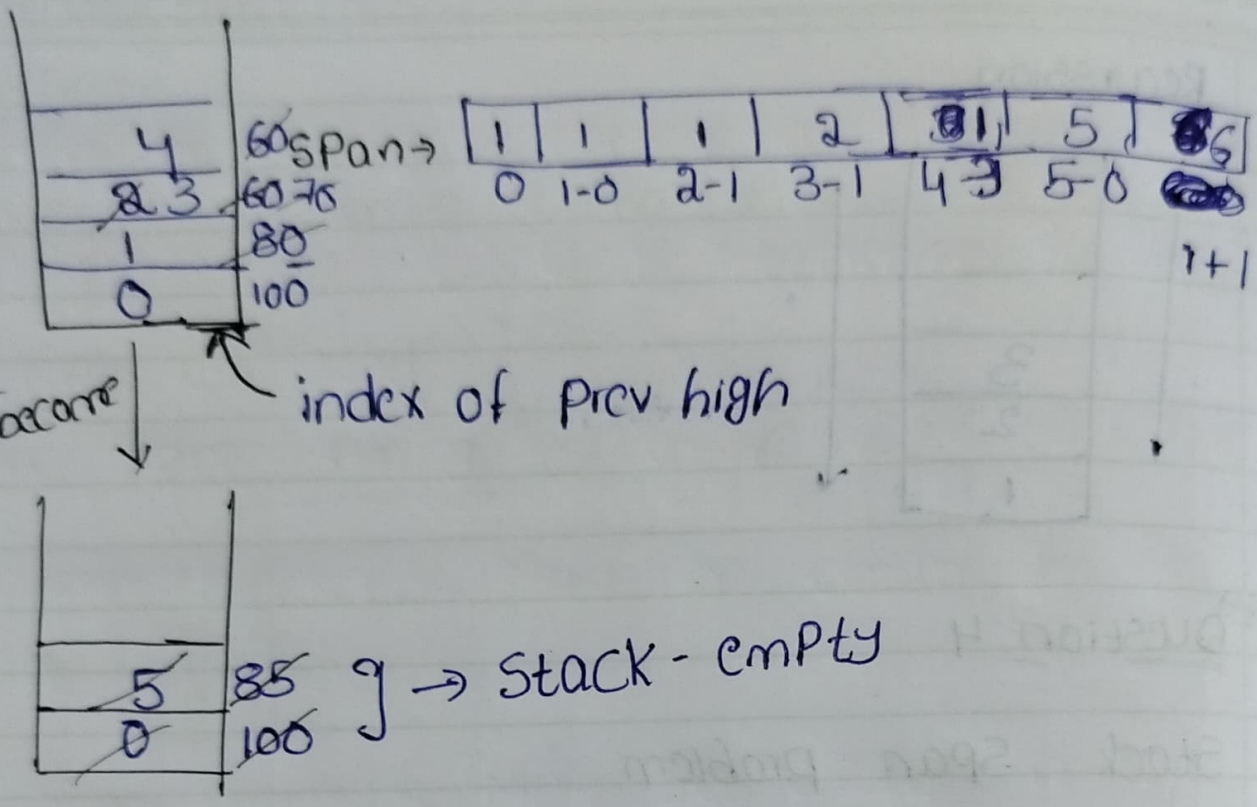
3 - 1 = 2

4 - 3 = 1

5 - 0 = 5

6 - 0 = 6

Teacher's Signature _____



25 Next Greater Element

The next greater element of some element x in array is the first greater element that is to the right of x in the same array

arr = [6, 8, 0, 1, 3]

next Greater = [8, -1, 1, 3, -1]

logic is very important we can use this in so, many problems

Date: .....

Expt. No.: ..... Page No.: .....

Brute force

Nested loop

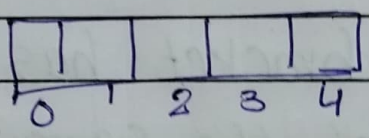
```
for(int i=0; i<arr.length; i++) { //outer loop
for(int j=i+1; j<arr.length; j++)) { //inner loop

    if(arr[i] < arr[j]) ?
```

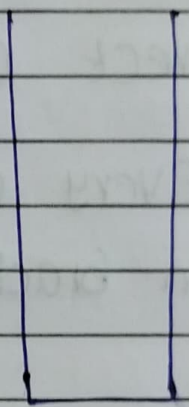
1st time

next Greater El TC → $O(n^2)$

Optimize TiMe complexity Using Stack

Approach arr = [6, 8, 0, 1, 3]

nextGreater[] = 



IMP

① while(!S.isEmpty() && Stack(top) <= arr[i])
S.pop()

② if Stack.isEmpty
nextGreater = -1

Note: ~~Code~~ ki ~~three~~ index p

else
nextGreater = S.peek()

③ S.Push(el)

Teacher's Signature _____

Valid Parentheses

Given a string s containing just the characters '(', ')', '{', '}', '[' & ']', determine if the input string is valid.

An input string is valid if:

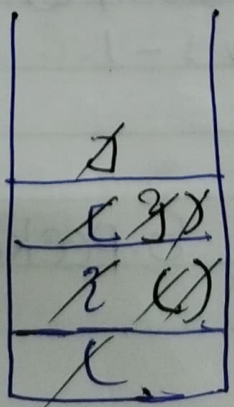
1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.
3. Every close bracket has a corresponding open bracket of the same type.

Ex:-

s = "()[]{}" ✓, s = "(]" ✗, s = "()" ✓, s = ")(" ✗

Approach

({ [ ] } ( ))



1) opening bracket.
S.Push()

2) Closing bracket.
↳ Stack → top
Pair ✓ True
Not Found False

Stack.isEmpty() ✓ True

code

```
for(int i=0; i<Str.length(); i++){
    ith → ch
    opening → S.push
    closing
        ↳ [redacted] → Pair ↔ S.peek()
                        S.pop()
        ↳ return false (INVALID)
}
S → Empty() → True
X → false
```

Duplicate Parentheses

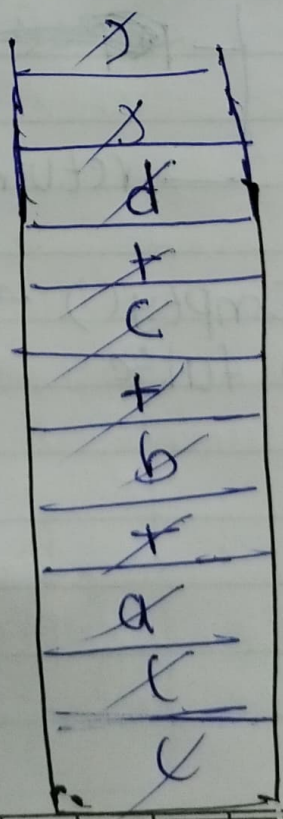
Given a balanced expression, find if it contains duplicate parentheses or not. A set of parentheses are duplicate if the same subexpression is surrounded by multiple parentheses
Return a true if it contains duplicates else return false.

Ex:- (((a+(b)))+(c+d)) → true

Ex:- ((((a)+(b))+c+d)) → true

Ex:- ((a+b)+(c+d)) → false

Ex:- (((a+b))+c) → true



Approach

Ex:- ((a+b)+(c+d))

false

count = ~~0~~ ~~1~~ ~~2~~ 3

Date: .....

Expt. No.: .....

Page No.: .....

① opening operator (+, -, *) operand (a, b, c)

S.push()

② Closing

count = 0 // items

try to find opening pair

(count<1) → Duplicate → True

3<1 ✗ False Duplicate not exist

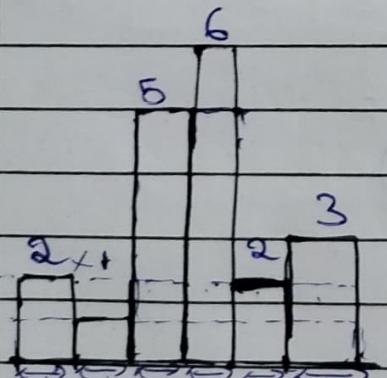
```
while(S.peek()=='('){
    S→pop
    count++
}
count<1→
S.pop()←'('
```

IMP 40-50 Advanced

Max Area in Histogram

Given an array of integers heights representing the histogram's bar height where the width of each bar is 1, return the area of the largest rectangle in the histogram

height = [2, 1, 5, 6, 2, 3]



Area = width × height

2 6 10 6 8 3 = 2 6 (10) 6 8 3 → hight

Ans → 10

Teacher's Signature _____

Approach

Area = height x width

↓ height[i]

↓ left, right

Note:-

For example h = [2, 1, 5, 6, 2, 3]

we take 5 → In this left side small (remove eliminate) right side (remove the small)

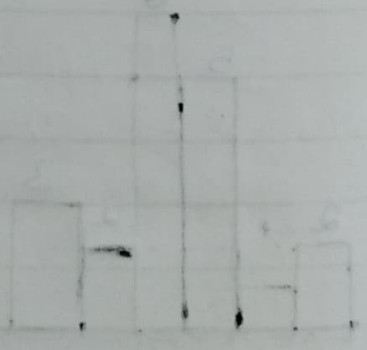
index (area = height x width

width

next Small (left) (i)

next Small (right) (j)

width = j - i - 1 → logic



Date: .....

Expt. No.: .....

Page No.: .....

→ For ex: h[index] = 6 - i^{th}

→ 4 - 3 = 1 → width

For ex: h[1] = j - i - 1

= 6 - (-1) - 1

= 6 + 1 - 1

= 6

h[5]

4 - 1 - 1 = 4 - 2 = 2 h - [2, 1, 5, 6, 2, 3]

(-1 -1 1 2 1 4) → index

nsl → | -1 | -1 | 1 | 5 | 1 | 2 | nsl → next Smaller left

nsr → | 1 | 6 | 2 | 2 | 6 | 6 | nsr → next Smaller right.

(1 n 4 4 n n) → index

Note: nsr → where a next smaller element is (1) next smaller element not their but you did't not take -1 instand of 6 → n

nsl → start with "3" next smaller element is (2) next smaller (1) next smaller element is (5)

Area = height x width width = (j - i - 1)

= 2 x (1 + 1 - 1)

= 2 x 1 = 2

Teacher's Signature _____