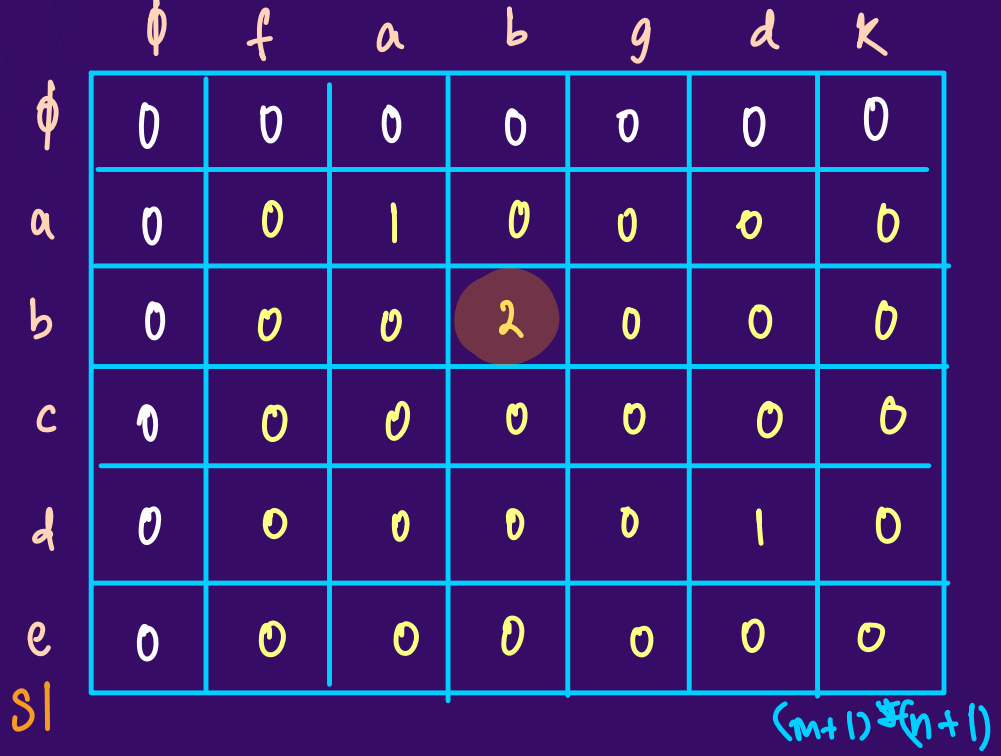
Ques:

Q : Longest Common Substring (Tabulation)



ans is not dp[m][n]

Ques:

Q : Longest Common Substring (Tabulation)

```
int m = a.length(), n = b.length();
int[][] dp = new int[m+1][n+1];
int max = 0;
for(int i=1;i<=m;i++){
    for(int j=1;j<=n;j++){
        if(a.charAt(i-1)==b.charAt(j-1))
            dp[i][j] = 1 + dp[i-1][j-1];
        else
            dp[i][j] = 0;
        max = Math.max(max,dp[i][j]);
    }
}
return max;
```

Ques:

↑ print LCS.

PW SKILLS

Q : Shortest Common Supersequence → String (length)

s1 = "abac" s2 = "cab"

LCS = ab

s CS = "cabac"

Find the length of shortest common supersequence of s1 & s2

- Think about LCS of s1, s2

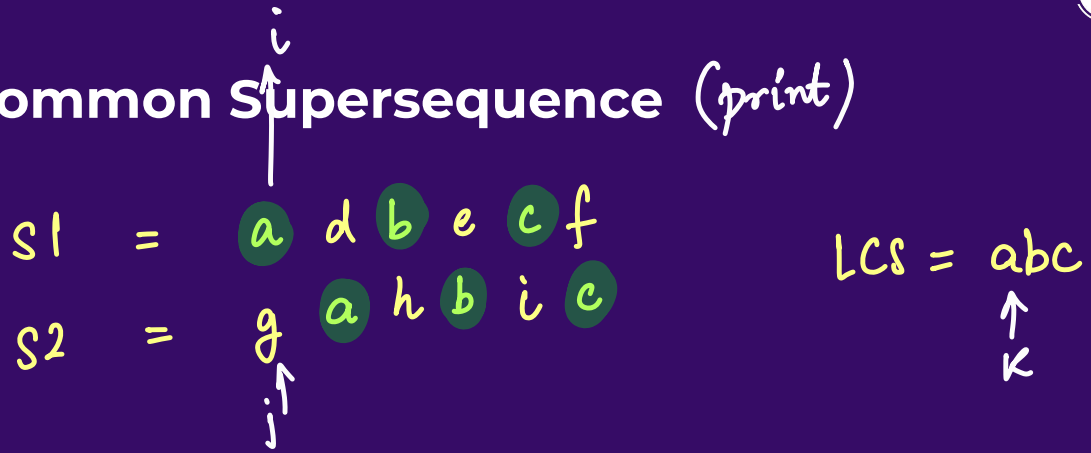
↓

LCS + s1 - LCS + s2 - LCS

[Leetcode 1092]

Ques:

Q : Shortest Common Supersequence (print)



SCS =

g a d h b e i c f

g a h d b i e c f

g a d h b e e c f

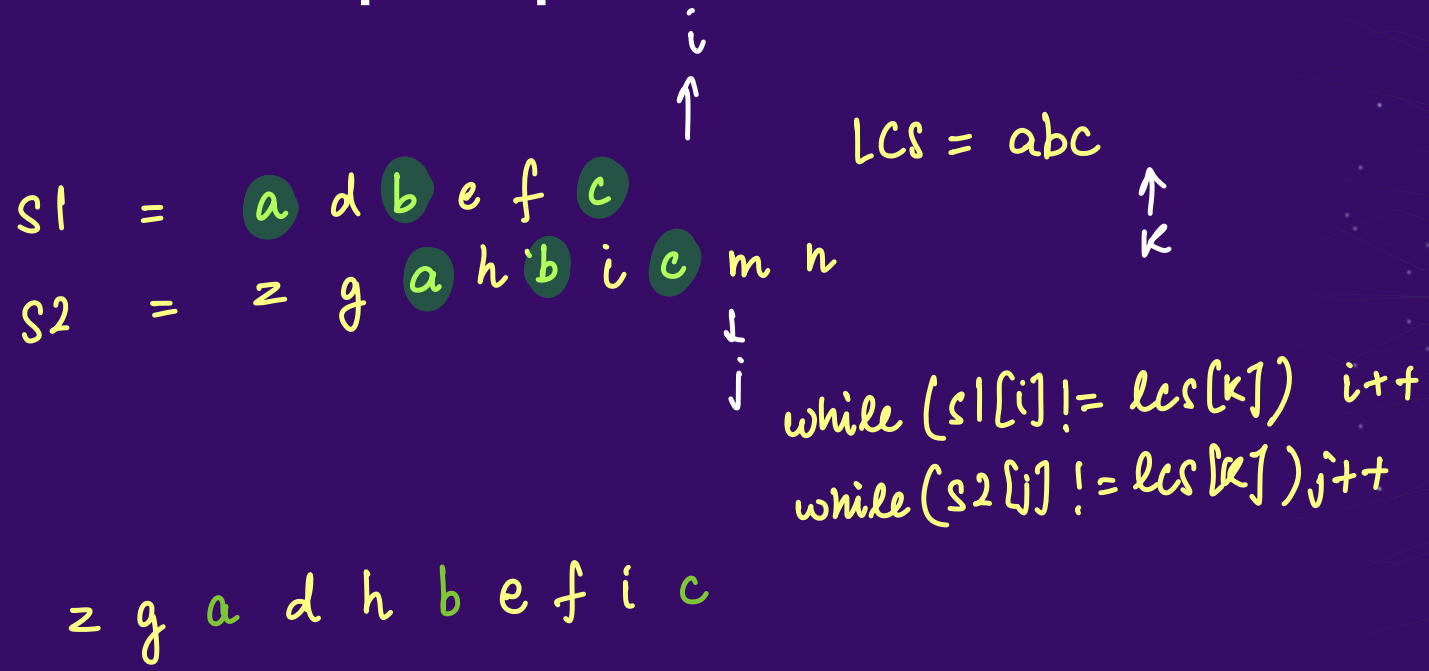
g a h d b i i c f

[Leetcode 1092]

Ques:

PW SKILLS

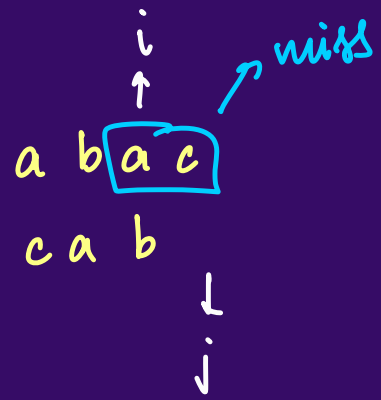
Q : Shortest Common Supersequence



[Leetcode 1092]

Ques:

Q : Shortest Common Supersequence

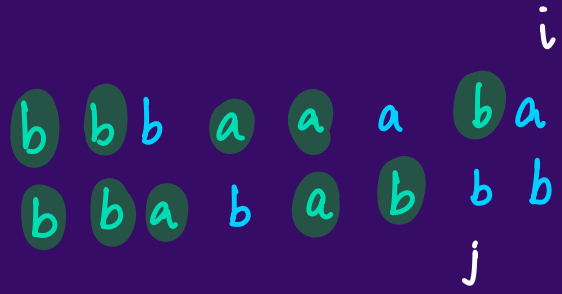


Ans = c a b a c

[Leetcode 1092]

Ques:

Q : Shortest Common Supersequence



LCS = b b a a b

K

SCS = b b b a b a a b b b a

b b b a a a b a b b b

[Leetcode 1092]

Ques:

PW SKILLS

Q : Shortest Common Supersequence

store LCS wali string

S1 = a d b e f c

S2 = g a h b i c

a d b e f c

g a h b i c

SCS = g a d h b i e f c

[Leetcode 1092]

Ques:

Q : Shortest Common Supersequence

```
public String shortestCommonSupersequence(String a, String b) {
        String lcs = LCS(a,b);
        int i = 0, j = 0, k = 0; // i->a, j->b, k->lcs
        String scs = "";
        while(k<lcs.length()){
            while(a.charAt(i)!=lcs.charAt(k)){
                scs += a.charAt(i++);
            }
            while(b.charAt(j)!=lcs.charAt(k)){
                scs += b.charAt(j++);
            }
            scs += lcs.charAt(k);
            i++; j++; k++;
        }
        while(j<b.length()){
            scs += b.charAt(j++);
        }
        while(i<a.length()){
            scs += a.charAt(i++);
        }
        return scs;
    }
```

[Leetcode 1092]

Ques: (Tabulation)

PW SKILLS

Q : Palindromic Substrings $s = gabcbab$

g, a, b, c, b, a, b

bcb, bab

abcba

Method-1 : Brute Force

↓

generate all substrings & check each substring

[Leetcode 647]

Ques:

$\sum k = \frac{k(k+1)}{2}$ $\sum k^2 =$

PW SKILLS

Q : Palindromic Substrings

For Ex : S = a b c b a $n=5$

a, ab, abc, abcb, abcba

$1+2+\cdots n + 1+2\cdots n-1 + 1+2+\cdots n-2$

$$\sum_{k=1}^{n} 1+2+3\cdots k = \sum_{k=1}^{n} \frac{k(k+1)}{2} \approx \sum k^2+k$$

$$\sum_{1}^{n} k^2 = \frac{n(n+1)(2n+1)}{6} = n^3$$

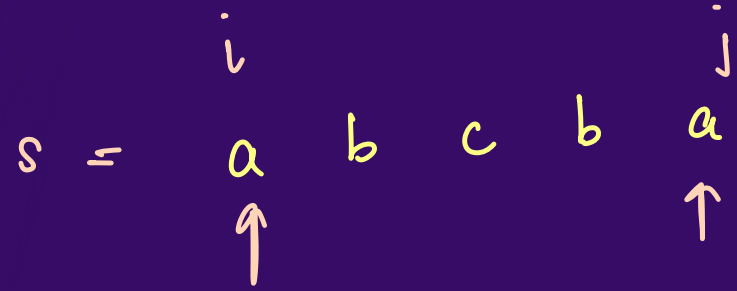
T.C. = $O(n^3)$

[Leetcode 647]

Ques:

Q : Palindromic Substrings

s = g a b c b a b (indices 0 1 2 3 4 5 6)



s[i] == s[j] && (i+1, j-1) substring is palindrome

[Leetcode 647]

Ques:

Q : Palindromic Substrings

	0	1	2	3	4	5	6
	g	a	b	c	b	a	b

2-length substring palindrome kab hoti hai?

A.S. = $O(n^2)$

T.C. = $O(n^2)$

i, j → i+1, j+1

	0	1	2	3	4	5	6
0	1	0	0	0	0	0	0
1	X	1	0	0	0	1	0
2	X	X	1	0	1	0	0
3	X	X	X	1	0	0	0
4	X	X	X	X	1	0	1
5	X	X	X	X	X	1	0
6	X	X	X	X	X	X	1

Every cell stores true/1 if substring from i to j is palindrome

[Leetcode 647]

Homework:

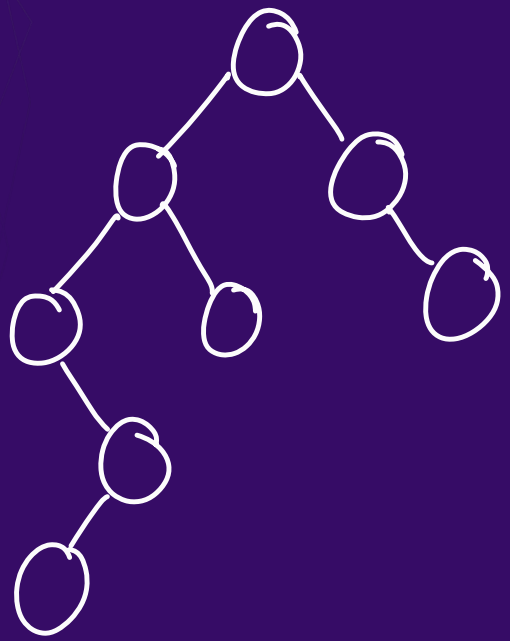
Q : Longest Palindromic Substring

[Leetcode 5]

Ques:

longest path

Q : Diameter of a Binary Tree



maximum = diameter

path length = n - 1

no. of nodes in the path

$n = 1 + \text{levels(left)} + \text{levels(right)}$

for a particular peak (node) ← $\text{dia} = \text{levels(left)} + \text{levels(right)}$

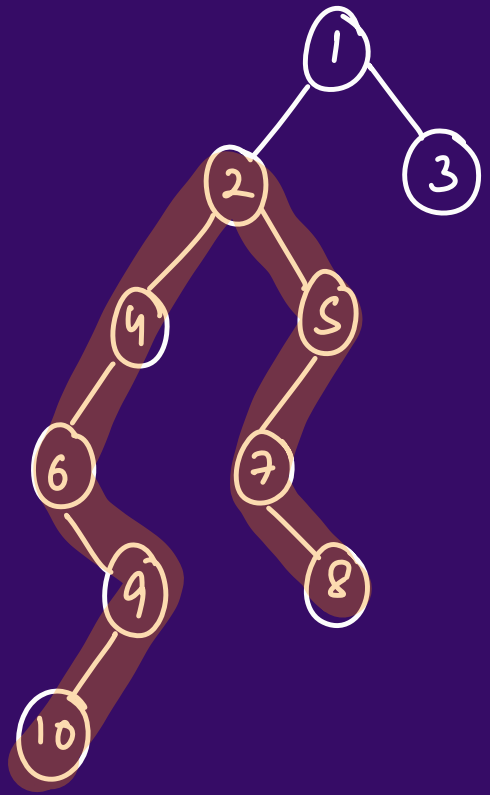
[Leetcode 543]

Ques:

dia = levels(left) + levels(right)

PW SKILLS

Q : Diameter of a Binary Tree



does not passes through the root

$O(n^2)$ T.C.

bcz for every node we are calling levels → O(n)

We can memoise using Hashmap

Map < TreeNode , Integer > dp;
node levels

[Leetcode 543]

Ques:

(Recursion + Memoization)

PW SKILLS

Q : Diameter of a Binary Tree

```
public int levels(TreeNode root, Map<TreeNode,Integer> dp) {
    if(root==null) return 0;
    if(dp.containsKey(root)) return dp.get(root);
    int leftLevels = levels(root.left,dp);
    int rightLevels = levels(root.right,dp);
    dp.put(root,1 + Math.max(leftLevels,rightLevels));
    return dp.get(root);
}
public int diameter(TreeNode root, Map<TreeNode,Integer> dp) {
    if(root==null) return 0;
    int myDia = levels(root.left,dp) + levels(root.right,dp);
    int leftDia = diameter(root.left,dp);
    int rightDia = diameter(root.right,dp);
    return Math.max(myDia,Math.max(leftDia,rightDia));
}
public int diameterOfBinaryTree(TreeNode root) {
    Map<TreeNode,Integer> dp = new HashMap<>();
    return diameter(root,dp);
}
```

T.C. = O(n)

A.S. = O(n)

[Leetcode 543]

Ques:

Q : Diameter of a Binary Tree

```java
public int levels(TreeNode root, int[] dia) {
    if(root==null) return 0;
    int leftLevels = levels(root.left,dia);
    int rightLevels = levels(root.right,dia);
    int path = leftLevels + rightLevels; // extra
    dia[0] = Math.max(dia[0],path); // extra
    return 1 + Math.max(leftLevels,rightLevels);
}
public int diameterOfBinaryTree(TreeNode root) {
    int[] dia = {0};
    levels(root,dia);
    return dia[0];
}
```

$T.C. = O(n)$

$A.S. = O(h)$

↓

recursive stack

[Leetcode 543]

Ques:

Q : Diameter of a Binary Tree (without memoization) DP

[Leetcode 543]

Homework:

Q : Balanced Binary Tree

[Leetcode 110]

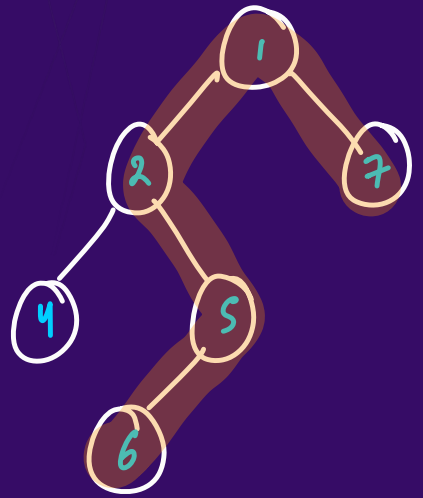
diameter/balance → levels

max path sum → ?

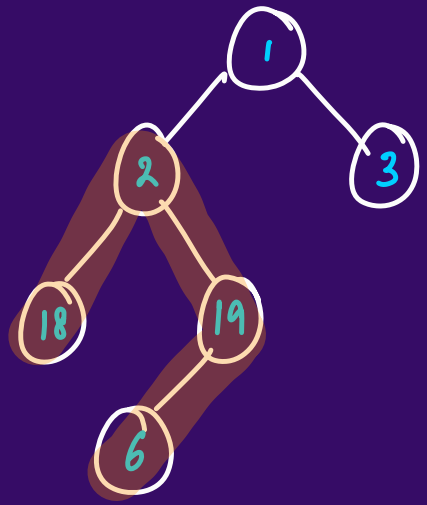
Ques:

Q : Binary Tree Maximum path sum

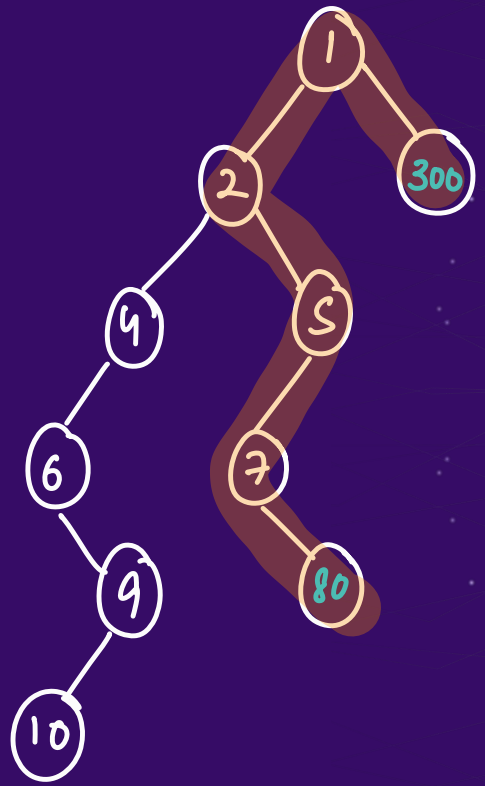
- Diameter of B.T.



6+5+2+1+7 = 21



18+2+19+6



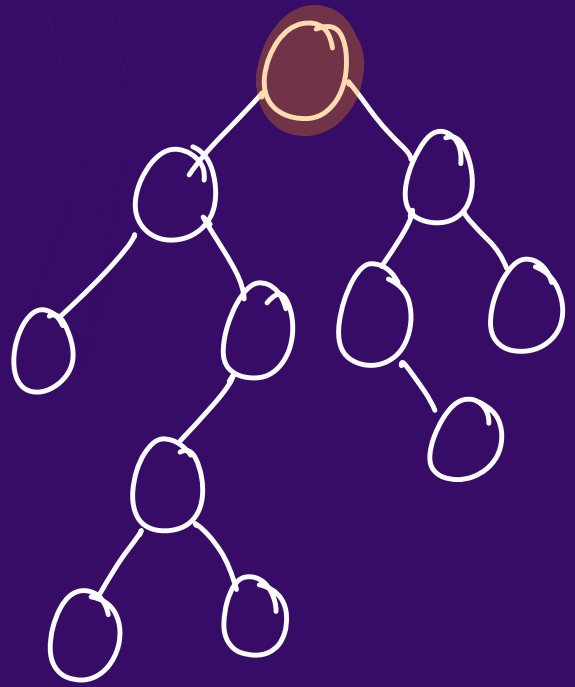
[Leetcode 124]

Ques: As of now I am considering leaf to leaf

PW SKILLS

Q : Binary Tree Maximum path sum

root -> leaf



pathsum = root.val + lineSum(left) + linesum(right)

↓

line sum ko 'maximise'

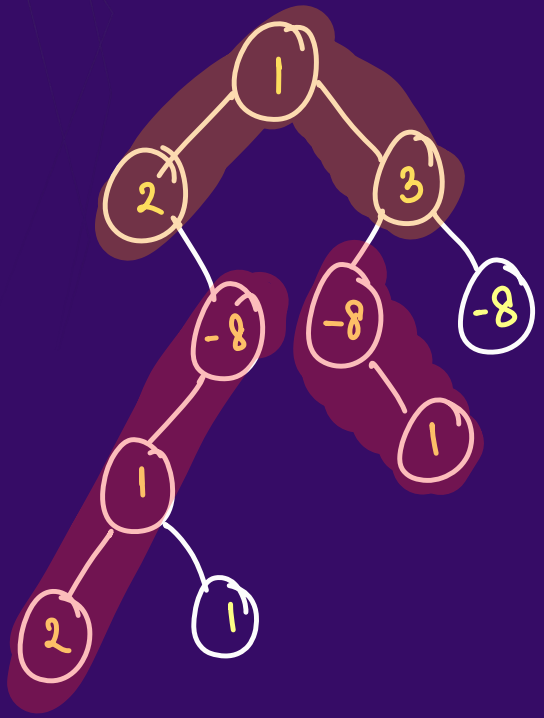
maxSum = max(sum, maxSum);

lineSum(root) = root.val + max(linesum(left), linesum(right))

[Leetcode 124]

Ques:

Q : Binary Tree Maximum path sum



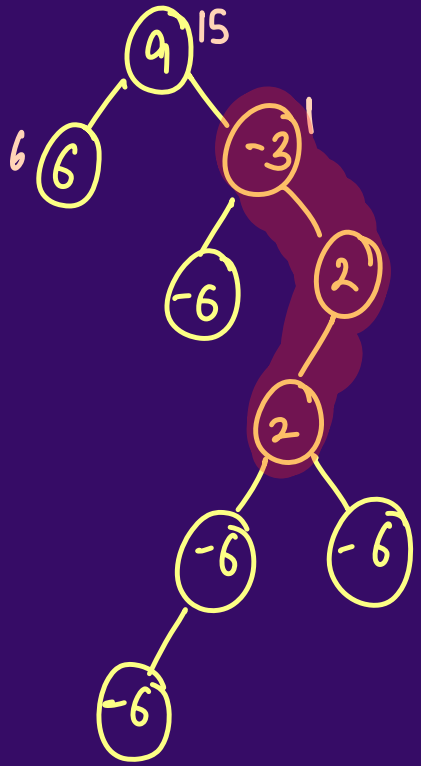
1 + 2 + 3 = maxSum

-3 → ans = -3

[Leetcode 124]

Ques:

Q : Binary Tree Maximum path sum



Out → 15
Exp → 16

[Leetcode 124]

Ques: dp!!

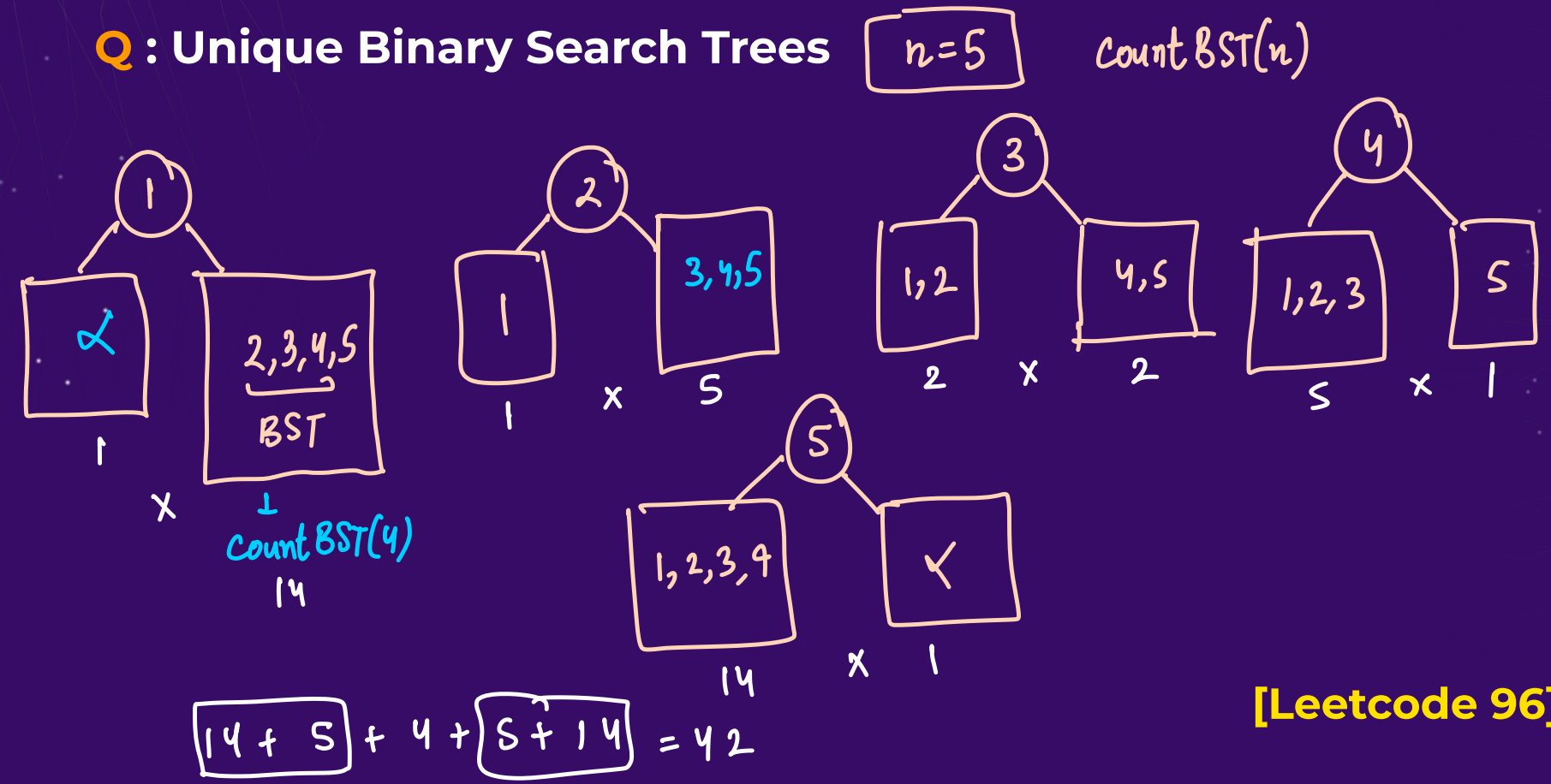
PW SKILLS

Q : Unique Binary Search Trees

[Leetcode 96]

Ques:

Q : Unique Binary Search Trees



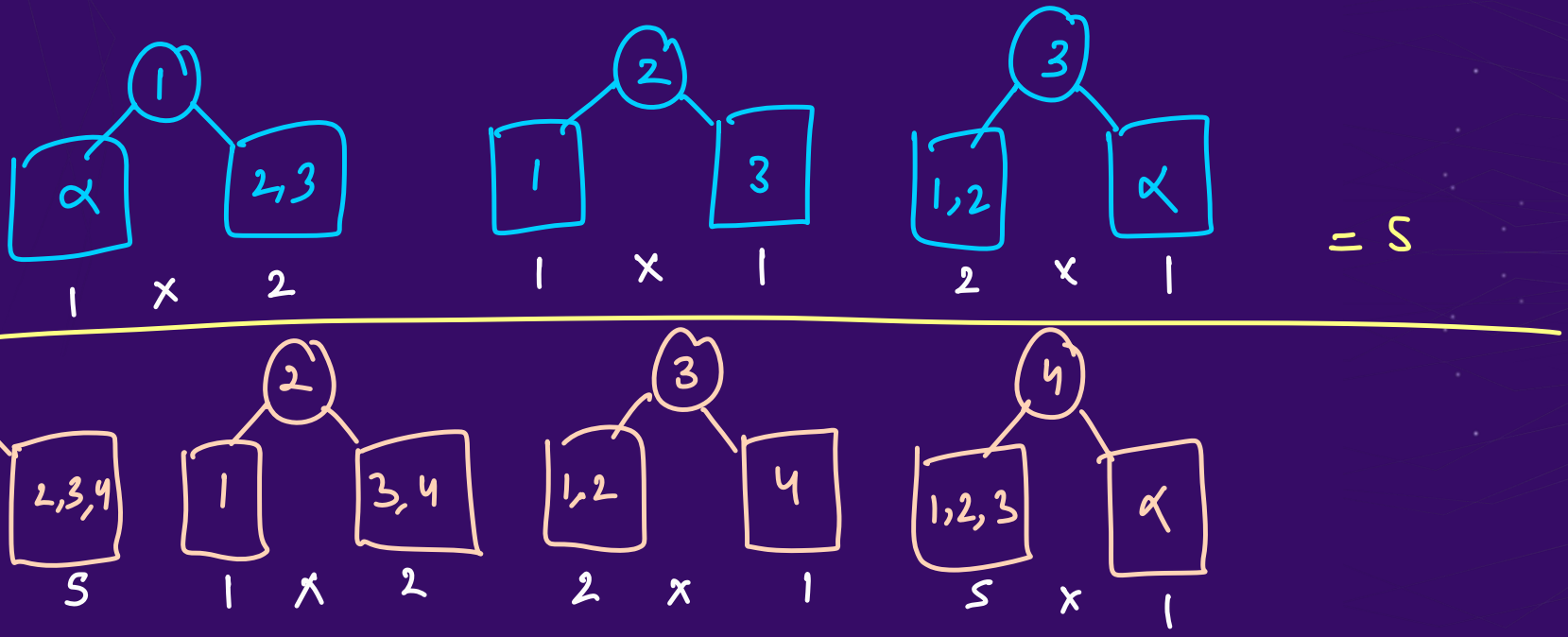
[Leetcode 96]

Ques: dp

0	1	2	3	4	5
1	1	2	5	14	42

PW SKILLS

Q : Unique Binary Search Trees (1D DP) (nested loop)



[Leetcode 96]

Ques:

Q : Unique Binary Search Trees

```
public int numTrees(int n) {
    if(n<=1) return 1;
    int[] dp = new int[n+1];
    dp[0] = 1; dp[1] = 1; dp[2] = 2;
    for(int i=3;i<=n;i++){
        for(int j=1;j<=i;j++){
            dp[i] += (dp[j-1]*dp[i-j]);
        }
    }
    return dp[n];
}
```

$T.C. = O(n^2)$

$A.S. = O(n)$

[Leetcode 96]

Ques:

Q : Longest Increasing Subsequence

arr = { 10, 9, 2, 5, 3, 7, 101, 18 }

1 1 1 2 2

arr = { 10, 9, 2, 5, 3, 7, 101, 18, 6 }

dp = { 1, 1, 1, 2, 2, 3, 4, 4, 3 }

[Leetcode 300]

Ques:

Q : Longest Increasing Subsequence

```java
public int lengthOfLIS(int[] nums) {
    int n = nums.length;
    int[] dp = new int[n];
    int max = 0;
    for(int i=0;i<n;i++){
        for(int j=0;j<=i-1;j++){
            if(nums[j]<nums[i])
                dp[i] = Math.max(dp[i],dp[j]);
        }
        dp[i] += 1;
        max = Math.max(max,dp[i]);
    }
    return max;
}
```

$T.C. = O(n^2)$

$A.S. = O(n)$

[Leetcode 300]

Ques:

Largest possible length of mountain array after removals

PW SKILLS

Q : Minimum Number of Removals to make Mountain Array

[Leetcode 1671]

Ques:

PW SKILLS

Q : Minimum Number of Removals to make Mountain Array

arr	2	1	1	5	6	2	3	1
dp1	1	1	1	2	3	2	3	1
dp2	2	1	1	3	3	2	2	1
	2 ✗	1 ✗	1 ✗	4	5	3	4	1 ✗

Normal LIS

Reverse LIS

[Leetcode 1671]

Ques:

PW SKILLS

Q : Minimum Number of Removals to make Mountain Array

	9	8	1	7	6	5	4	3	2	1
dp1	1	1								
dp2	9	8								
	9	8								

[Leetcode 1671]

Ques:

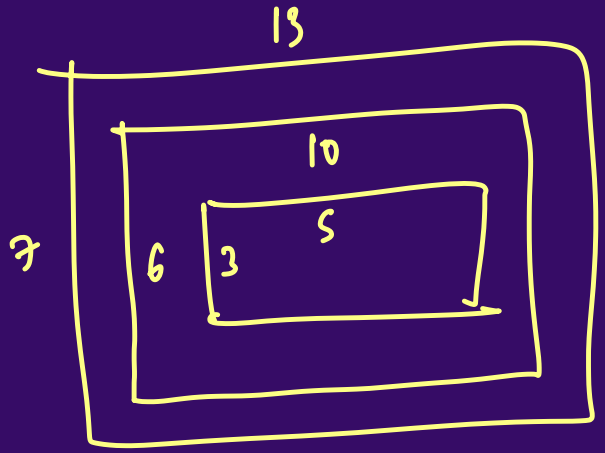
PW SKILLS

Q : Russian Doll Envelopes

w h

arr = { {3,5}, {7,13}, {6,10}, {8,4}, {6,7} {7,11} }

ans = 3



(a,b) (c,d)

(a,b) can get inside (c,d) if & only if

a < c && b < d

[Leetcode 354]

Ques:

PW SKILLS

Q : Russian Doll Envelopes

Hint → 1 Sorting (Custom) (Comparable)

Hint → 2 LIS

arr = { {3,5}, {7,13}, {6,10}, {8,4}, {6,7} {7,11} }

Sort(row) = {3,5} {6,7} {6,10} {7,11} {7,13} {8,4}

sort(col) = {{8,4}, {3,5}, {6,7}, {6,10}, {7,11}, {7,13}}

[Leetcode 354]

Ques:

PW SKILLS

Q : Russian Doll Envelopes

arr = { {3,5}, {7,13}, {6,10}, {8,4}, {6,7} {7,11} }

Sort(w) = {3,5} {6,7} {6,10} {7,11} {7,13} {8,4}

Sort(w: asc
h: des) = { 3,5} {6,10} {6,7} {7,13}, {7,11}, {8,4}

[Leetcode 354]

THANK YOU