ocanneu wini cant

182777	16/06/2023
QI	Longest increasing subsequence
	i/p→ {5,8,3,2,1,9,73
	Total subsequences possible = 2n. Filter out
Mille	the increasing subsequence and we have to return the longest length subsequence.
Time .	Lease of a ment of a self of the little for
	Incheasing subsequence -> [5,8,9] -example of
	increasing subsequence
LVEN 4.	Le periodicional application of all plants for
	Approach bullier a shalling rum and and
-	[0,1,0,3,2,3]
	include exclude
D. Pari	Ancient Canaline Cana
	CoJ L CoJ L
	i e i e
	[0,1] [0] [1]
	i/(e i/(e i//e
	Z X X
	X  [0]  X  [0]  [1]  [0]  [
Note >	We need to include a particular element only
	when it is bigger than previous element.
1 .	
4	Code

```
int solve Rec (vector Kint > & nums, int cour, int prev)
      // Base case
      if (curr) = nums size())
            retwin os
      // include call
      int include = 0;
     // 1st element or previous element smaller
      If (prev = = -1 | I nums [cwr] > nums [prev]){
          include = 1 + solve Rec (nums, curit, curi)
     //exclude call
     int exclude = 0 + solve Rec (nums, cury+1, prev);
     int ans = max (include, exclude);
     return ansi
                                          remains
                                          same
// Top down approach
int solve Top Down (vector < int) & nums, int curs
int prev, vector <vector <int>> & dp) {
     // Base case
     if (cwur >= nums · size())
            return 0;
    // Step 3: Check if ans already exists
    if (dp [cwv ][prev+1] [= -1)
           retwin dp[curr][prev+1];
    Minclude call
    int include = 0;
   if (prev /= -1 // nums [cwur] > nums [prev]
        include = 1 + solve Top Down (nums)
          Cwur +1, cwur, db);
   /exclude
```

```
int exclude = 0 + solve Top Down (nums, court),
      prev, dp)
     // Step 2: Store one in dp array
     // prev + I as prev = - I can be there and prev+1 = 0.
     dp[cwv][prev+1] = ansi
                                           valid index
     return of [cwr][prev+1];
  // Bottom up approach
  int solve Tab (vector <int> & nums) {
      int n = nums size()
      // Step 1 : Create of array
      vector (vector (int)) dp(n+1, vector (int) (n+1,0)
     //Step2: Observe base case of top-down - Handled
     1/Step3: Reverse flow of top-down
      for (int cur = n-1; (wur >=0; cur --) {
        for (int prev = cwm = 1; prev > = -1; prev --) {
            Minclude call
            int include = 0 > 18.5)
           if (prev == -1 | Inums [curi] > nums [prev]){
               include = 1 + dp (curr+1) (curr+1);
          Mexclude 3 dinoi han simon
          int exclude = 0+dp(cuvv+1)(prev+1);
          dp[awr][prev+1] = max (include, exclude);
      return aprostos;
* Space optimization possible or not?
    Yes it is possible.
```

. 41).	Create 2.1D arrays.
	Create 2,1D arrays.  vector Kint > cworRow (n+1,0);
	vector (int) next Row (n+100);
2)	Replace dp [cww +1] with next Row []
L []	and replace of [cours] with cour Pow []
3)	Shift - next Row = curr Row - going upwards
	DO III I
	DP with binary search
1 6	arr 7 [10,9,2,5,3,7,101,18]
17111	WOI 7 [ 103 13 , 2 , 5 , 5 , 7 , 101 , 16 ]
1 3	1 = 0 → [10]
	$i=1 \rightarrow [9]$ (Overwrite)
11-	$i = 2 \rightarrow [2]  (0 \text{ verwrite})$
	$i=3 \rightarrow [2,5]$ (Include)
	$i = 4 \rightarrow [2,3]$ (Dverwrite)
(-	1=15 → [2,3,3,7] (Include)
1000	i=67 [2,3,7,101] (Include)
	$L=7 \rightarrow [2,3,7,18] \rightarrow Jenath = 4$
	(Overwrite)
Note-	Overwrite and include is to be done.
	1 + 1 1 1 1 2 1 1 1 1 1 2 2 2 2 2 2 2 2
(51.1	Code
	°0 + C - 1
	int solve Optimal (vector (int) 4 arr) {
	$if(\omega w \cdot size() = = 0)$
	retwin 0;
	vector <int>ans;</int>
	ans. bush - back (avr [0]);

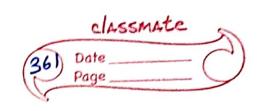
Scarined with Cam

	for (int i=1; i <avr.size(); i++){<="" th=""></avr.size();>
	if (avr [i] > ans bock ())
	ans bush - back (ann [17) )
	else { //overwrite
	// find index of just bigger element
	int index = lower-bound (ans-begin(),
	ans end (), ary (i) - ans begin ();
	ans [index] = arm [i];
	3
	3
	return ans. size();
	3 Aller and a second and a second as a sec
	TC = O(nlogn)
*	Russian doll approach
	7 height
	OUR - {[5,4],[6,4],[6,7],[2,3]}
	width "
	2/
	Arrange them in increasing order of width of width is same, then take higher height Simply apply LIS solution on the width now.
	width is same, then take higher height Simply
	apply LIS solution on the width now.
	(S) MODEL : NY BOOKER (S)
	(2,3] Chipach Mark
	[5,4]
	[6,7]
	[6,4]
	Now apply LIS on (2,5,6,6) and hence here 3 is the answer.
	thence there 3 is the answer.

Scarineu with Cail

<u>Q2</u>	Max height by stacking cuboid.
	1/b- [50,45,20]
	[95,37,53]
	[45,23,12]
	First sort. (We need maximum height.
	[20,45,50]
	[37,53,95]
	[12,23,45]
	(15)(2)(10)
2)	Now sort according to the width
	W l h
	[12,23,45] $(0,00)$
	[20,45,50]
	[37,53,95]om dan Nob masa A
	t).
3)	Now Simply apply LIS.
	Code
464	they to restry the roal or med grands
Admi	int max Height (vector (vector (int >> & cuboids) {
: N .	//Sort every array for (auto &a: Cuboids)
	sort (a.begin(), a-end());
	//Sort 2D avray
	Sort (cuboids begin () , cuboids end ())
	// Apply LIS logic
	int ans = SpaceOpt (cuboids);
	3 return ans i
	J PROPERTY OF THE PROPERTY OF





```
bool check (vector (int)a, vector (int)b){
       // Compare length, width & height
if (b[o] <= a[o] && b[i] <= a[i] &&
          b[2] <= a[2])
                return true;
      else
               retwin false;
Changes in LIS code (Space optimization)
if (prev = = -1 && check (nums [curr], nums [prev]
    // height to be added instead of I
    include = nums [cwr][2] + next Row [cwr+1];
```