Final Contest & Mock Interview.

Career Eligibility ⇒ Mock Interviews.
+
Full Syllabus Contest

Company X → DSA ✓, LLD ✓, SQL ✓

Expert Mock Interview → 2 questions → 1 Easy, 1 Easy-Medium (60 mins)

↓ 30 days of last class

+

Contest ⇒ 13th March (Live) ✓ | 14th — Weekend (R1) ✓ | 1 Week R2 ✓

(___)

KnapSack Problems

N Objects → Profit Loss Value ⇒ V[i]
→ Weight ⇒ W[i]

A bag with a given capacity C.

Select object s.t. total sum of selected object weight $\leq C$ & sum of profit/loss value is max/min.

Fractional KnapSack

Q: Given N cakes with their happiness & weights

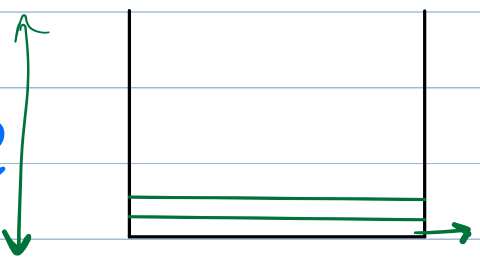
Visit the bakery with a bag of weight capacity C.

Find the max. happiness that can be kept in the bag. (Cakes can be divided)

$N = 5$

$C = 40$

	0	1	2	3	4
H =	3	~~8~~	~~10~~	2	5
W =	10	~~4~~	~~20~~	8	~~15~~ - 6 = 9
Parts =	10	4	20	8	15
Happiness/Part =	0.3	2	0.5	0.25	0.33
		1	2		



30

Rem Cap = ~~30~~ ~~26~~ 6

$Hap = \lfloor 8 + 10 + \frac{\cancel{5}^{1}}{\cancel{15}_{3}} \times 6^{2} \; 2 \rfloor = 20$

$\Rightarrow$ Greedy solution.

Sol^n

1) Sort objects wrt $H[i]/W[i]$

T.C. = $O(N \log N)$

2) Select the cakes in decreasing order & remaining capacity.

T.C. = $O(N)$

T.C. = $O(N \log N)$

---

0 - 1 KnapSack (Objects cannot be divided)

→ Take it / leave it

Q Toy Shop

Bag with capacity = 7 Kg

N Toys ⇒

$$\begin{array}{lccccc} & 0 & 1 & 2 & 3 \\ \text{Happiness} \rightarrow & [1, & 3, & 5, & 6] \\ \text{Weight} \rightarrow & [2, & 3, & 4, & 5] \end{array}$$

Select the toys to maximize the total Happiness value of the toys in the bag.

C = 7

Sol^n

0 1 2 3

Happiness → [1 , 6 , 9 , 12]
Weight → [2 , 3 , 4 , 5]

H/W = 0.5   2   2.25   2.4

Ans = **13**

Greedy fails ⇒ All poss.

1) **EOC** ⇒ Purchase (Select) or Not Purchase (Reject)

2) **State** (index, remaining capacity)

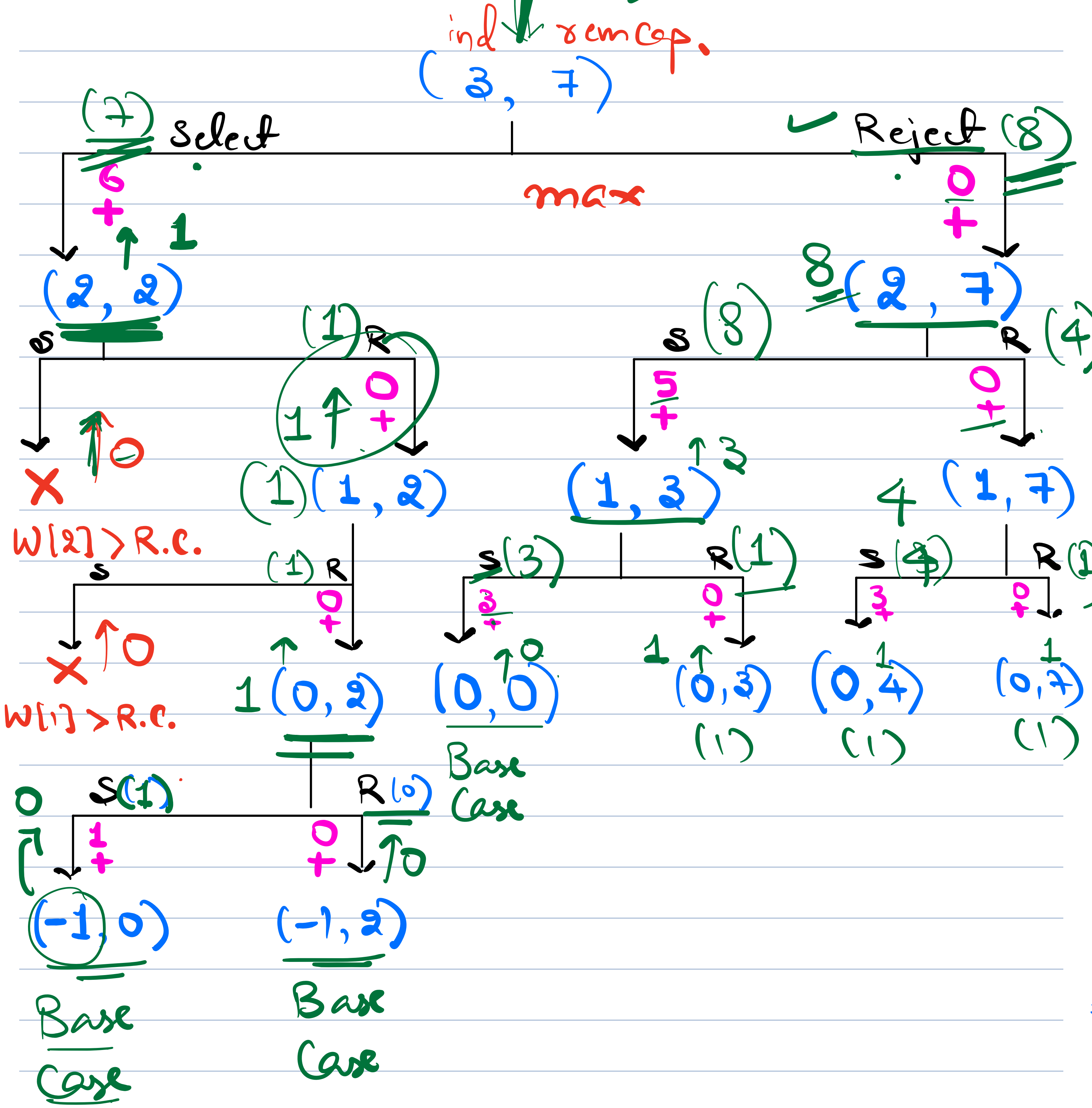
maxHappiness (i, c) ⇒ Max Happiness objects ⇒ [0, i] with capacity c

$$\begin{array}{lc} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \text{Happiness} \rightarrow & [\begin{array}{cccc} 1, & 3, & 5, & 6 \end{array}] \\ \text{Weight} \rightarrow & [\begin{array}{cccc} 2, & 3, & 4, & 5 \end{array}] \end{array}$$

c = 7

7

8



3) Recurrance Rel^n

$$\text{Happiness}(\underset{ind}{i}, \underset{rem}{j}) \begin{cases} \xrightarrow{S} H[i] + \max \text{Happiness}(i-1,\ j - W[i]) \\ \quad \max \\ \xrightarrow{R} \max \text{Happiness}(i-1,\ j) \end{cases}$$

Cap

$$\text{Happiness}(\underset{\text{rem. weight.}}{\overset{\text{index}}{i}}, j) = \max\left(H[i] + \text{Happiness}(i-1, j-W[i]), \; \text{Happiness}(i-1, j) \right)$$

4) Which state ??

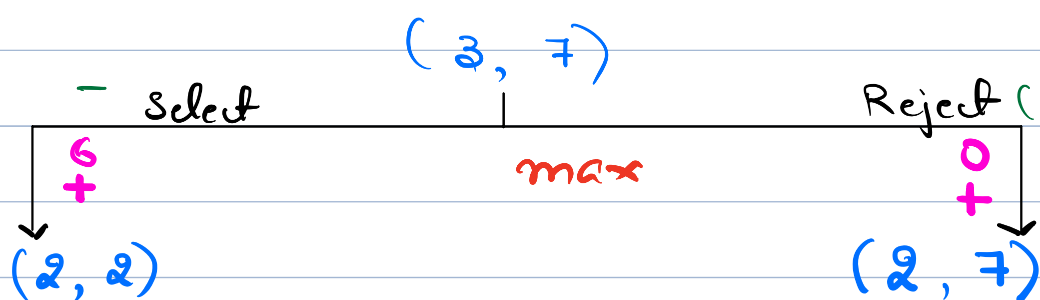
$$\max \text{Happiness}(\underset{}{\overset{[0, N-1]}{N-1}}, C)$$

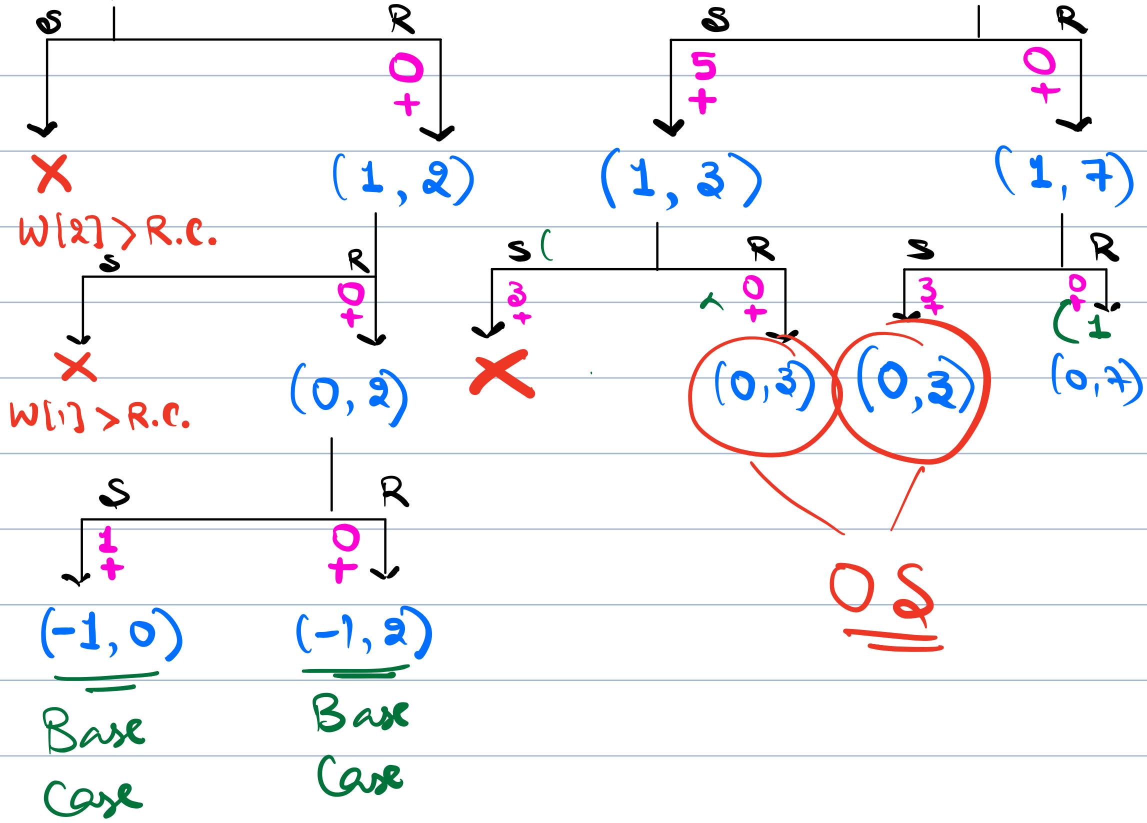
$$T.C. = O(2^N)$$

Do we have overlapping subproblem?

	0	1	2	3
Happiness →	1	3	5	6
Weight →	2	4	4	5

C = 7





DP Solution

$i == -1$

$C == 0$

$\{-1[0, N-1]$ 1 to C

$DP[N][C] \Rightarrow N \times C$

index

$DP[N+1][C+1]$

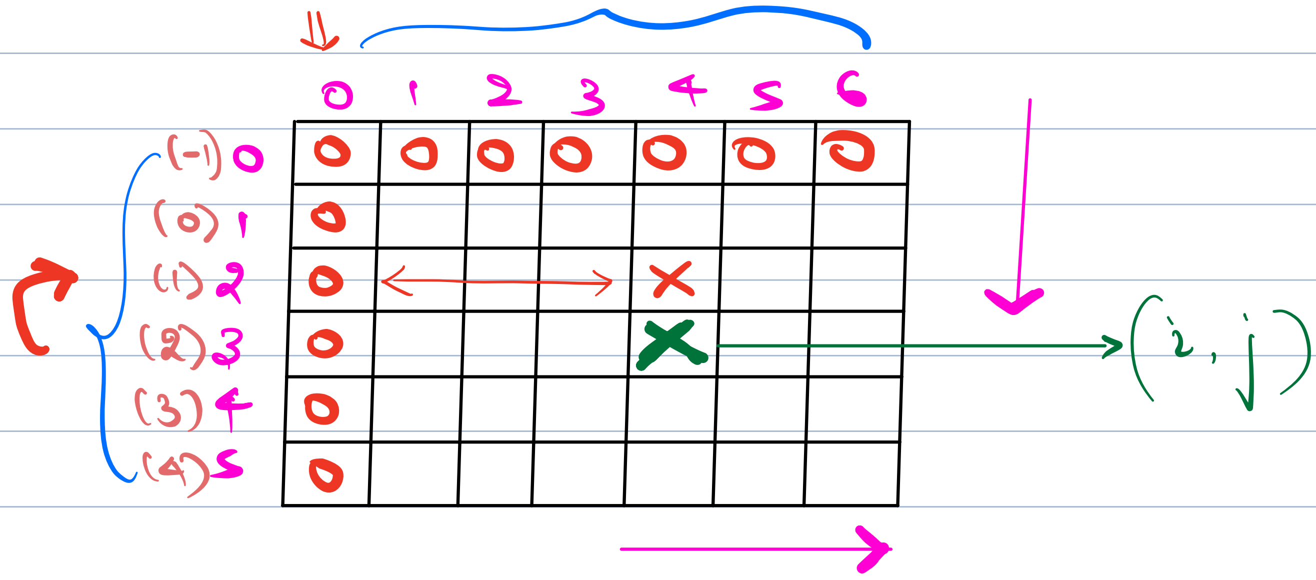
index

$$\text{Happiness}(i, j) = \max\big(H[i] + \text{Happiness}(i-1, j-W[i]),\ \text{Happiness}(i-1, j)\big)$$

rem.

weight.

DP[N+1][C+1]



Code

```
DP[N+1][C+1] = {0}

for (i=1; i<=N; i++) {
    for (j=1; j<=C; j++) {
        if ( w[i-1] <= j ) {
            DP[i][j] = max( DP[i-1][j],
                            H[i-1] + DP[i-1][j - w[i-1]] )
        }
        else {
            DP[i][j] = DP[i-1][j];
        }
}
```

}

```
return DP[N][C];
```

T.C. = $O(N \times C)$

S.C. = $O(N \times C)$

0/N KnapSack (Unbounded)

Unbounded → Select an object any no. of times

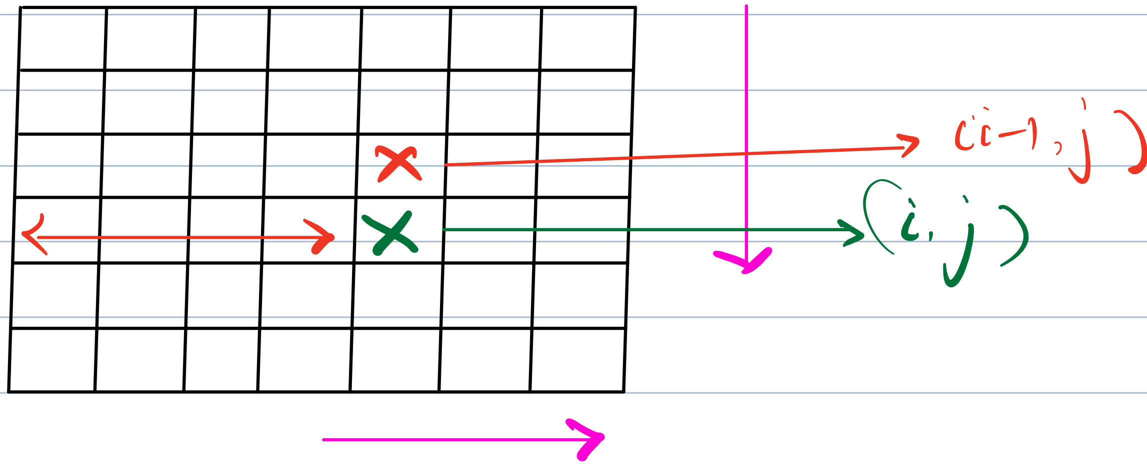
0/1 — S / R } Element is processed

0/N —
- Reject → Element is processed.
- Select → Not processed as we can select it again

Reject → DP[i-1][j]

$DP[i][j]$ — max

select → $H[i] + DP[i][j - W[i]]$



T.C. = $O(N \times C)$

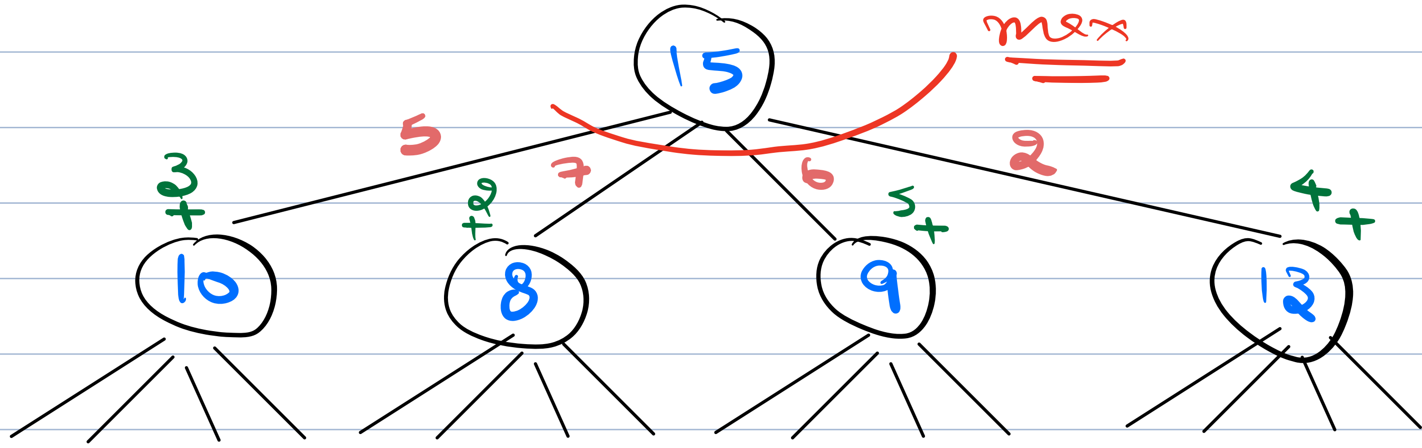
S.C. = $O(N \times C)$ ⇒ 2D DP

Q. Can we solve this as a 1D DP problem??

State ⇒ ~~index~~, rem. capacity.

H ⇒ [3, 2, 5, 4]   N = 4

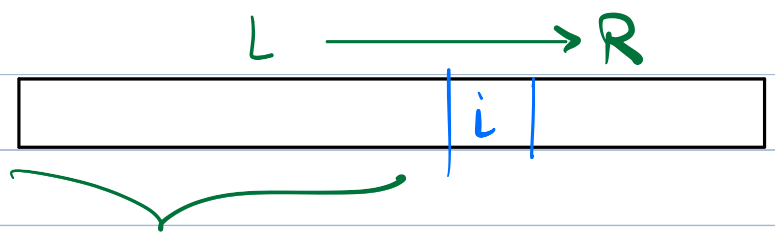
W ⇒ [5, 7, 6, 2]   C = 15



1D DP

DP[i] ⇒ Max Happiness in a bag with capacity i.

$$DP[i] = \max\left(\forall_{\text{objects}\,(W[\text{object}] \le i)} \left(H[obj] + DP[\underbrace{i - W[obj]}_{< i}] \right) \right)$$



Code

```
DP [C+1] =
          DP[0] = 0
for (i=1; i<=C; i++) {   // Capacity
    maxH = 0;
    for ( j=0; j<N; j++) {   // Element
```

```
        if (W[j] <= i) {
            happ = H[j] + DP[i - W[j]];
            maxH = max(maxH, happ);
        }
    }
    DP[i] = maxH;
}
return DP[c];
```

T.C. = $O(N \times C)$
S.C. = $O(C)$

$$\begin{array}{cccc} & 0 & 1 & \\ A = & [\,1, & 30\,] & \\ W = & [\,1, & 50\,] & \end{array}$$

Ans = 100

Q Dungeon & Princess

Given ⇒ Matrix of size $N \times M$
Each cell of the matrix represents a room.

In every room
- → Dragon ⇒ $M[i][j] < 0$
- → Healer ⇒ $M[i][j] > 0$
- → $M[i][j] = 0$ ⇒ Empty.

Prince will start from (0,0) with initial health H.

Prince $\rightarrow$ Dragon $\Rightarrow$ H + M[i][j] ($\downarrow$)

$\rightarrow$ Healer $\rightarrow$ H + M[i][j] ($\uparrow$)

If $(H \leq 0)$ $\Rightarrow$ Prince is dead.

Find the min initial health of prince s.t. the prince can reach the princess alive.

P

(H = ??)

-3	2	4	-5
-6	5	-4	6
-5	-7	5	-2
2	10	-3	-4

$\rightarrow$ Princess

H = 1

Move:

$(i, j) \rightarrow (i, j+1)$

$\downarrow$

$(i+1, j)$

Happiness → [1 , 3 , 5 , 6]
Weight → [2 , 4 , 4 , 5]

indices: 0, 1, 2, 3

$c = 7$

$1 + 0$
$= 1$

5 × 8

		0	1	2	3	4	5	6	7
(-1)	0	0	0	0	0	0	0	0	0
✓ (0)	1	0	0	1	1	2	2	3	3
⇒ (1)	2	0							
(2)	3	0							
(3)	4	0							

$1 + \cancel{1}$
$\cancel{2}$

$$DP[i] = \max\left(DP[i-1][j],\ H[i-1] + DP[i][j - W[i-1]]\right)$$

DP[N+1][C+1] = {0}

```
for (i=1; i<=N; i++) {
    for (j=1; j<=C; j++) {
        if ( W[i-1] <= j ) {
            DP[i][j] = max( DP[i-1][j],
                H[i-1]+ DP[i][j- W[i-1]]
        }
        else {
```

DP[i][j] = DP[i-1][j];

STAR

~~[illegible]~~

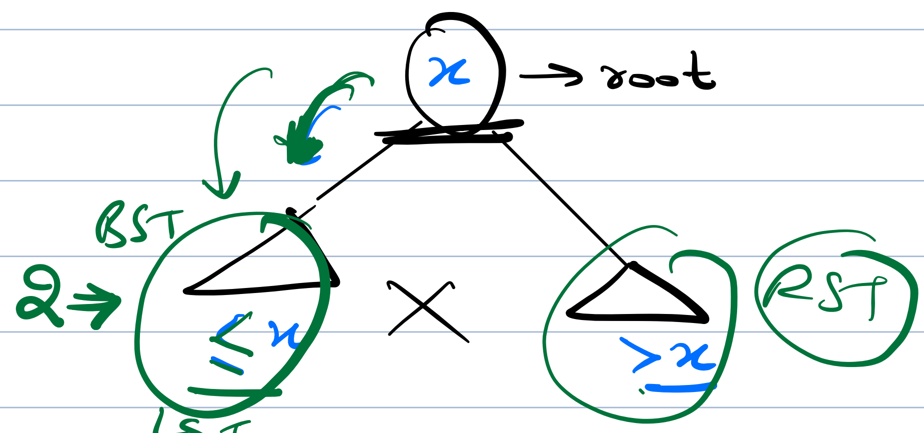
Star Tar Art

Result

N unique no.

form BST

[1, 2, 3, 4, 5]



	L	R
1	0	4
2	1	$\boxed{3}$
3	2	2
4	3	1
5	4	0

$$C(N) = c(0)\times c(4) + c(1)\times C(3) + c(2)\times c(2)$$
$$+ c(3)\times c(1) + c(4)\times c(0)$$

Catalan Series $\Rightarrow$

N pair of paren $\rightarrow$ Valid Seq

N

Catalan

Mathematics

[1, 2, 3] | [3, 4, 5]