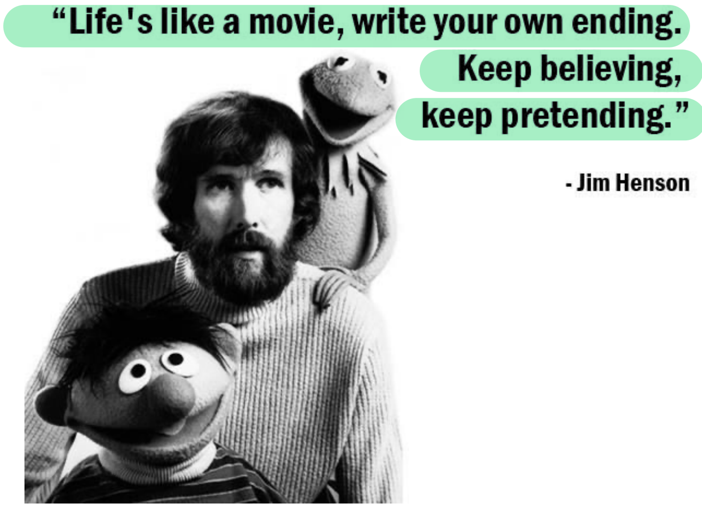


Good
Evening

Content

01. Unbounded knapsack
02. Rod cutting problem
03. Coin sum permutation
04. Coin sum combination
05. 0-1 Knapsack 2 {Idea for this}

Q Given N toys with their happiness & weight
Find max total happiness that can be kept in a bag with weight $\leq W$

Note → toys can't be divided

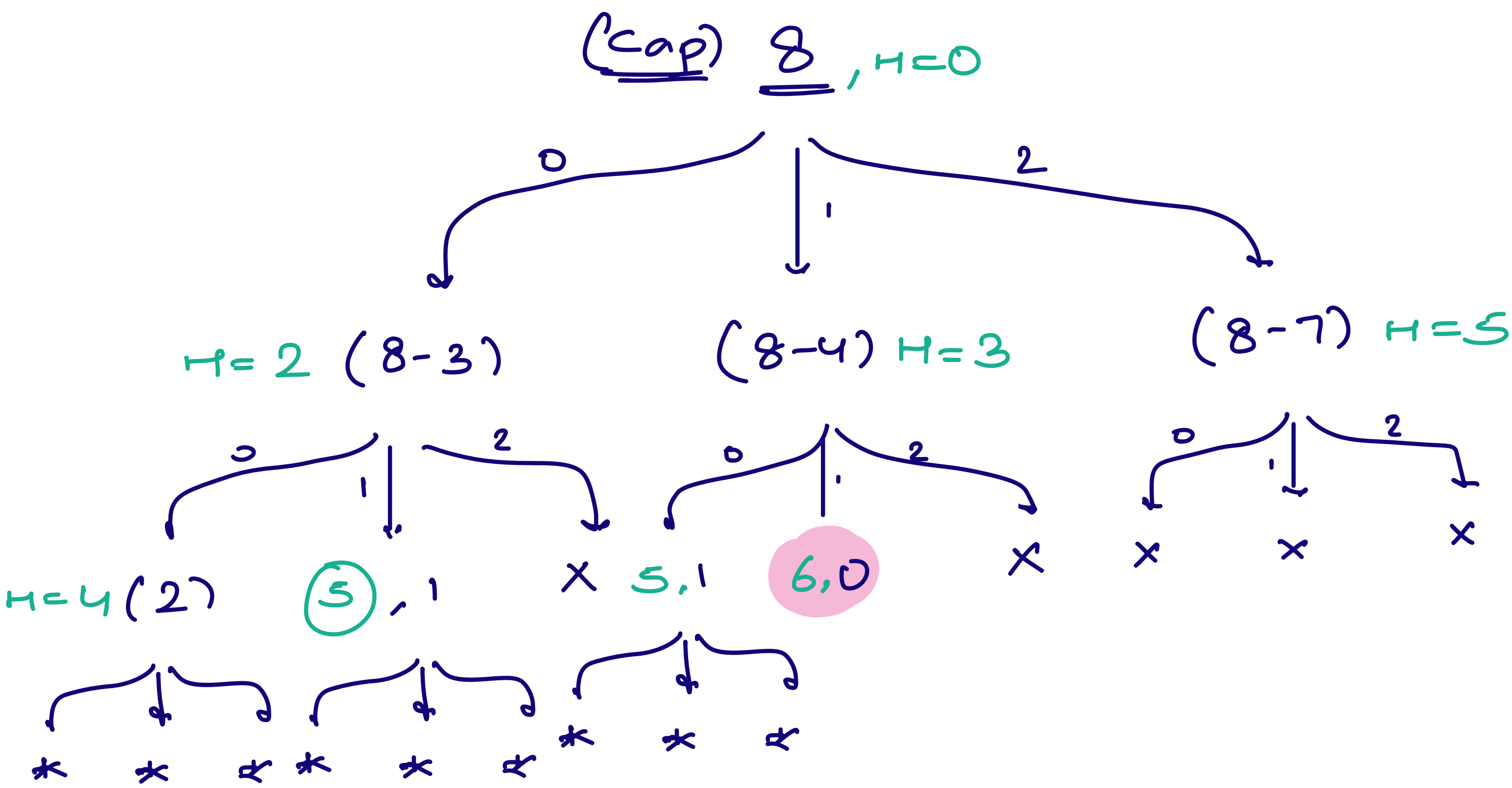
Note → Infinite supply of toys

$N = 3$

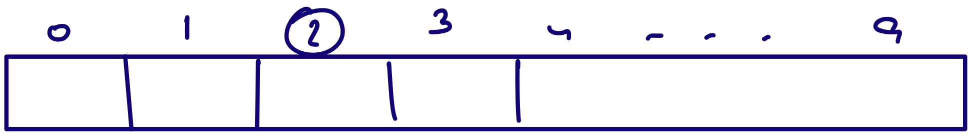
$W = 8$

	0	1	2
h[] →	2	3	5
wt[] →	3	4	7

Ans = 6 → pick 1st toy 2 times



dp[] → size = W+1



dp[i] = max happiness we can generate
with bag capacity as i.

dp[0] = 0; # for all i, dp[i] = 0

```
for (i=1; i≤W; i++){
    for(j=0; j<N; j++){
        if( i ≥ wt[j]){
            dp[i] = max( dp[i], h[j]+ dp[i-wt[j]]);
        }
    }
}
return dp[W];
```

TC : O(W * N)

SC : O(W)

	0	1	2
h[] →	{ 2	3	5 }
wt[] →	{ 3	4	7 }

0	1	2	3	4	5
0	0	0	2	3	3

cap = 5

0 | 1 | 2

$\underbrace{h[0]} + dp[5-3]$ $\qquad$ $h[1] + \underbrace{dp[5-4]}$ $\qquad$ x

$2 + 0$ $\qquad$ $3 + 0$

$= 2$ $\qquad$ $\Rightarrow 3$

Q Given a rod of length N & an array of length N.

A[i] → price of i length rod

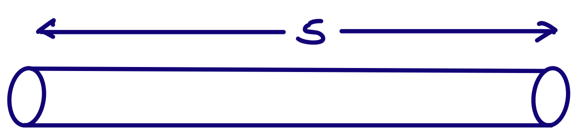
find the maximum value we can obtain by selling the rod

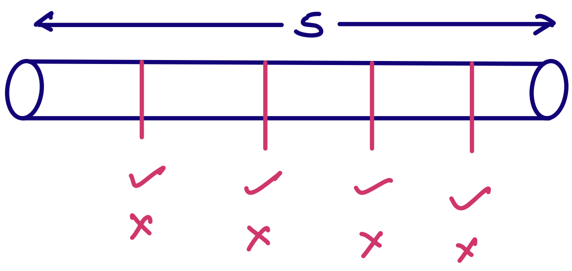
Note → we can sell the rod in pieces.

N = 5

$A = \{0\ 1\ 4\ 2\ 5\ 6\}$

indices: 0 1 2 3 4 5





$\#$ ways $= 2^4 = \underline{16}$

sold length	Total value
5	6
1 + 4	1+5 = 6
2 + 3	4+2 = 6
1 + 1 + 3	1+1+2 = 4
2 + 2 + 1	4+4+1 = 9
1 + 1+1+1+1	1+1+1+1+1 = 5

Limited cap = Length of given rod

weight of item = diff len of rod

value [i] = A [i] (price)

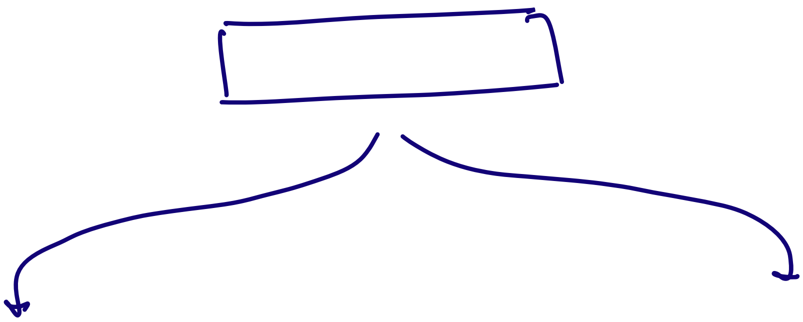
dp[i] → Max profit you can get by selling a rod of length i

N = 5

A[] = { 0, 1, 4, 2, 5, 6 } → prices of rods of i length

0 1 2 3 4 5 ← len of rod

0	1	2	3	4	5
0	1	4	5	8	9
	1 len	2len	1+2	2+2	



cut	best price we can get if we are selling our rod of len 1
1	1*

$1 + 1$

$\Rightarrow 2$

cut	
2	0*

$4 + 0$

$\Rightarrow 4$

len = 3

cut	rem rod leng
1	2*

1 + 4

⇒ 5

cut	rem rod length
2	1

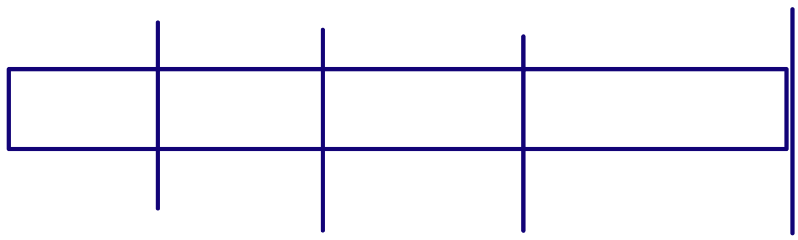
4 + 1

⇒ 5

cut	rem = 0
3	

2 + 0

= 2



len = 4

cut	rem rod len
1	3

1 + 5

⇒ 6

cut	rem rod
2	2

4 + 4

= 8

cut	remrod
3	1

2 + 1

= 3

cut	ren
4	0

5 + 0

= 5

len = 5

cut	rem	cut	rem	cut	rem	cut	rem	cut	rem
1	4*	2	3	3	2	4	1	5	0

1 + 8 = 9

4 + 5 = 9

2 + 4 = 6

5 + 1 = 6

6 + 0 = 6

TC : $O(N^2)$

SC : $O(N)$

```
for ( i = 1 to N )
    for ( j = 1 to i )
        dp[i] = max(dp[i], A[j] + dp[i-j]);
    }
}
return dp[N];
```

j = cut

Max profit we can make for len i-j

10 : 07 pm → 10 : 17 pm

Coin change permutations

<u>Q3</u> In how many ways, can sum be equal to N by using coins given in the array.
One coin can be used multiple times

Ordered selection $(x,y) \neq (y,x)$

$K = 5$

$A = \{3, 1, 4\}$

$\{1, 4\}$ $\{4, 1\}$ $\{1, 1, 1, 1, 1\}$

$\{3, 1, 1\}$ $\{1, 3, 1\}$ $\{1, 1, 3\}$

cap = 5

weight = $\{3, 1, 4\}$

dp[i] = No. of permutation with which i sum can be achieved

0	1	2	3	4	5
1	1	1	2	4	6
—	1	1 1	— 3 1 1 1	1 3 3 1 1 1 1 1 4	

2

1

1

(1)

3

3 | 1

[0] | [2]

No. of ways to achieve sum 0 = 1

No. of ways to achieve sum 2 = 1

— 3

$\underbrace{1\ 1\ 1}$

4

3 | 1 | 4

[1] | [3] | [0]

No. of ways to achieve 1 = 1

No. of ways to achieve 3 = 2

No. of ways to achieve 0 = 1

<u>1 3</u>

— 3 1

1 1 1 1

— 4

5

3 | 1 | 4

[2] | [4] | [1]

ways to achieve 2 = 1

ways to achieve 4 = 4

ways to achieve 1 = 1

1 1 3

1 3 1

3 1 1

1 1 1 1

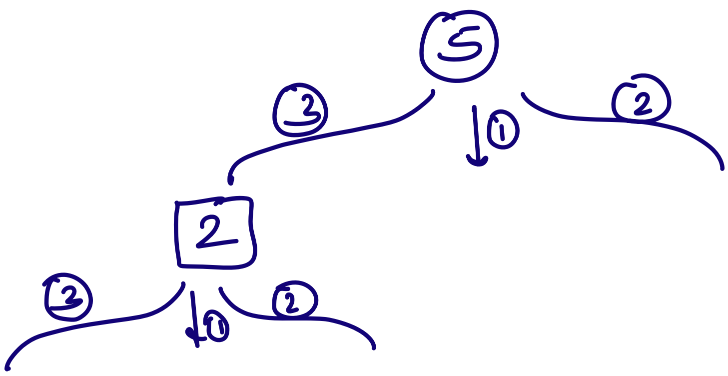
1 4

```
dp[0] = 1;                int [] dp = new int [K+1];

for (i=1; i≤K; i++){
|   for (j=0; j<n; j++){
|   |   if (ar[j] ≤ i){
|   |   |   dp[i] = dp[i] + dp[i - ar[j]];
|   |   }
|   }
}
return dp[N];
```

TC: (n * K)

SC: O(K)




```
ans = 0
for (j=0; j<N; j++)
|   if (ar[j] ≤ K){
|   |   ans += coinchange(arr, K - ar[j]);
}   }
```

* Coin sum combination

Unordered selection $(x,y) = (y,x)$

$N = 5$ ✓

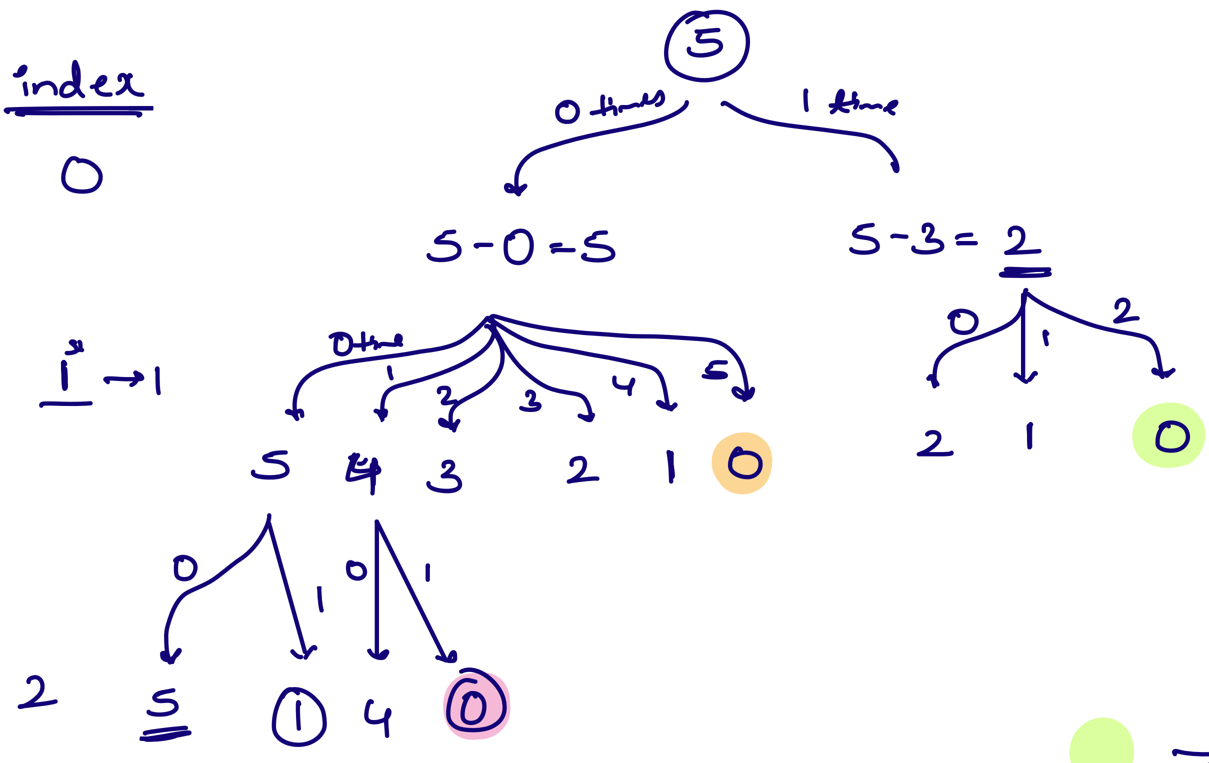
$A = \{3, 1, 4\}$

$\{1,4\}$ $\{1,1,3\}$ $\{1,3,1\}$

$\{4,1\}$ $\{3,1,1\}$ $\{1,1,1,1,1\}$

Ans = 3

* Criteria → Go with array order



→ 3 1 1

→ 1 1 1 1 1

→ 1 4

dp[i] = # ways to get sum = i by selecting coins
from L to R

```
int[] dp = new int[K+1];
    dp[0] = 1;

for ( j=0 ; j<n ; j++){
    for ( i=1 ; i ≤ K ; i++){
        if ( ar[j] ≤ i ){
            dp[i] = dp[i] + dp[i - ar[j]];
        }
    }
}
return dp[N];
```

N = 5

A = {3, 1, 4}

0	1	2	3	4	5
1	1	1	~~1~~ 2	3	~~2~~ 3
—	—1	11	—3	31	311
			111	1111	11111
				—4	14

0/1 knapSack 2

Given N items each with a weight & value, find max value which can be obtained by picking items such that total weight of all items $\leq$ K

Note 1 :- Every item can be picked at max 1 time

Note 2 :- We can not take a part of item

Constraints

$1 \leq N \leq 500$

$1 \leq val[i] \leq 50$

$1 \leq wt[i] \leq 10^9$

$1 \leq W \leq 10^9$

$TC = O(N * W)$

$= 500 * 10^9$

$= 5 * 10^{11}$ {TLE}

$$dp[N][W] = \underbrace{\text{max value}}_{500 * 50 = 25000}$$

wt

$$\underbrace{dp[N][value]}$$ → min weight require to achieve val

500 * 25000

= 12500000

$= \underbrace{1.25 * 10^7}$

```
for ( i = val ⟶ 0 )
|   if ( dp[N][i] ≤ W )
|          return i ;
}
```