→ Lesser mistake sunshine/ Hotstar

N: coins | 0 | 2 | 3 | 4 | 5 | 6 | 7
N = 8 : | 2 | 4 | 9 | 6 | 10 | 13 | 11 | 14

K = 23 | 23 | K = 40

path 1 : coins = [0-6] K = 3 + 14 = 23
Lesser 1 : Coins = [0-6] K = 40

Off Knap Sack

1) Problem is NOT T/F
Given N coins check if we can get sum K.

2) # ways to get sum
Given N coins check if we can get sum

3) # min no. of coins
min coins

Dp state:
dp[i][j] : using first i coins can we get sum j or not.

Dp state:
Dp[i,j] = ——, number of ways get sum j

Dp state:
Dp[i,j] = ——, min coins to get sum j

Dp_Choices
dp[i][j] = (dp[i-1][j] || dp[i-1][j - A[i]])
j >= A[i]

0-1 dp

dp[i][j] = (dp[i-1][j] + dp[i-1][j - A[i]])
j >= A[i]

dp[i][j] = min (dp[i-1][j] , dp[i-1][j - A[i]] + 1)
j >= A[i]

unbounded knap
dp[i][j] = dp[i-1][j] || dp[i][j - A[i]]

unbounded
Replace [i-1][j - A[i]] with [i][j - A[i]]

Replace [i-1][j - A[i]] with [i][j - A[i]]

Dp Table [1 based index]
dp[N+1][K+1]
base cases: 1 based index
Base: i == 0 // Do we have any coins?
∀ j = 1 : dp[0][j] = False
dp[0][0] = True;
Final dp[N][K]

1 Based, i == 0 / No coins
dp[N+1][K+1]
∀ j=0 : dp[0][j] = 0
dp[0][0] = 1
Final ans → dp[N][K]

1 Based, i == 0 / No coins
dp[N+1][K+1]
∀ j=1 : dp[0][j] = ∞
All values = ∞ / INT_MAX — overflow
∀ j=0 : dp[0][0] = 0
Ocean max value will become 0

Dp Table [0 based]
dp[N][K+1]
i == 0
∀ j=1 : dp[0][j] = False
dp[0][0] = True
dp[0][A[0]] = True if A[0] <= K
Final ans: dp[N-1][K]

// // ocean

→ unbounded; that Coin will change

[1 Based indexing]
1) Given N array elements, check if we can divide entire array into equal parts

Ex: {1, 3, 4, 2}
{1, 4} {3, 2}

Ex: {1, 3, 3, 3}
{ {S1} {S2} } N/2 N/2 → S2 {All remaining elements}

1) Hint: if sum % 2 → False
→ [0-1] knapsack + {S1, S2, (Total)/2}

2) Number of ways we can distribute [all array elements] into 2 subsets such that $S_1 - S_2 > K$ (S_1) (S_2)

$S_1 + S_2$ = (Array Sum)
$S_1 - S_2 = K$
$2S_1$ = Array Sum + K

{ $S_1 \geq$ }
Total sum = 7
K = 3
(7 + 3) → {even number}

If (Array Sum + K is even)
$S_1 = \frac{\text{Array Sum} + K}{2}$ → How many ways we can get S_1
If (Array + K is odd) → [0-1] Knap 2nd question
→ return 0

3rd) Given N array elements / Distribute all array elements, such that, their abs($S_1 - S_2$) is min

Ex: 3 2 4 7 6 3
{3, 3, 7} {4, 7, 6}
{3, 4, 7, 3, 7} {7, 6}
12 13
abs( ) > 1
abs($S_1 - S_2$) = 1

Ex: 2 2 10
{2, 2} {10}
→ 6

Total sum → 10
Minimum | S_1 | S_2
0 | 50 | 50 → dp[N][50] ✓
2 | 49 | 51 → dp[N][49] ✗
4 | 48 | 52 → dp[N][48] ✗
6
8
{50 - 1 × 2}

$S_1 + S_2$ = 101
S_1 | S_2
50 | 51 : 1 dp[N][50] ✗
49 | 52 : 3 dp[N][49] ✗
48 | 53 : 5 dp[N][48] ✗

Note: K = Sum/2
dp[N][K], we are also calculating, dp[N][K-1], dp[N][K-2]

K = Sum/2
for (i = K; K >= 0; K--) {
if (dp[N][K]) {
S_1 = K
S_2 = Sum - K
return (S_2 - S_1)
$S_2 > S_1$ → $S_1 + S_2$ = Sum, S_2 - Sum = S_1 [S_2 >> Sum/2] ✗ [$S_2 \geq S_1$]
Min diff
}
}
N, Sum
TC → Dp[N][Sum/2]

Hard
Given an array of N Elements, +ve.
→ 1) You can take any array element & flip it sign
2) You can take as many array elements as you want to flip it
3) Finally array sum is min non negative integer of {K(min): difference only} >= 0

{ Every element will + / − }
S_1 — All the signs | S_2 — All negative
sum (S_1) − sum (S_2)
All +

A[] = { 5, 4, 3, 7, 6 }
{5, −4, −3, −7, −6} Sum = 12 | {5, 3, 6, −4, −7, −7} −14 −18 | {3, 3, 2, 5, 1, −4} 12 −12 → 0

[0-∞] Knap / unbounded knapsack → {Dp}
→ We can pick value every year
& array we want

N = 5 : 3 4 7 6 9
K = 17
→ diff :
{ } 23 × 7, 13 ✓
→ 3 Solutions
Variety : they we don't want cover
Same thing repeats → 2 [ ]

→ coins : 1 2 sum = 3 (Prod)
Knapsack → 4
Dice : { 1 1 } { 1 2 } { 2 1 } → 2

using dice logic coinage not of using +*
using dice logic ever can dollar
→ # min no. coin needed : ✓
→ value given can possible or not

1) Bind
Finding → 17/27 — 23/27

2) Submarine sum
3) Replace TLE / change parameter
4) [1 2 3] 0 1 2 3

2) Question
→ 1 Limb