Madhan Kumar M S
Abhishek Sharma
amit khandelwal
Anuj chandil
Balaji S K
Bhavesh Rathod
Burhan
Dewnash
Gagan Kumar S
Hemant Kumar
Nikhil Pandey
Purusharth A
Rajat Sharma
Rajendra
Sanket Giri
Saurabh Ruikar
Shani Jaiswal
sharath r
shilpa mamillapalli
Shradha Srivastava
SHREYA GUPTA
Sneha L
Sridhar Hissaria
Subhashini
SUBHRANIL KUNDU
Suyash Gupta
Vimal Kumar
Yugesh v

AGENDA:

— Interview Problems

Important Events

— Full syllabus contest
15th May

— Mock Interview
self Book

Past → Time & Backlog Management ***
{Must watch} 5th May

A = 1 2 3 4 5

subarray — 1 2 3 ✓
3 2 4 ✗

subset — {1} {2} {3 4 5}
{2 3 5}
{4 2 3}

Target sum

You are given a set of non-negative integers and a target sum. The task is to determine whether there exists a subset of given set whose sum is equal to target sum.

A ⟶ 3 34 4 12 5 2 sum = 9

ans = true {4, 5}
{3, 4, 2}

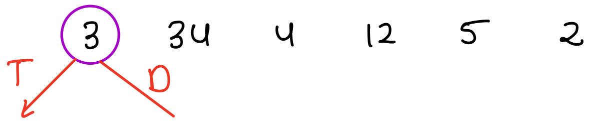
BF

⟶ Generate all the subsets ⟶ 2^N

Iterate to each subset and check if any subset == target sum

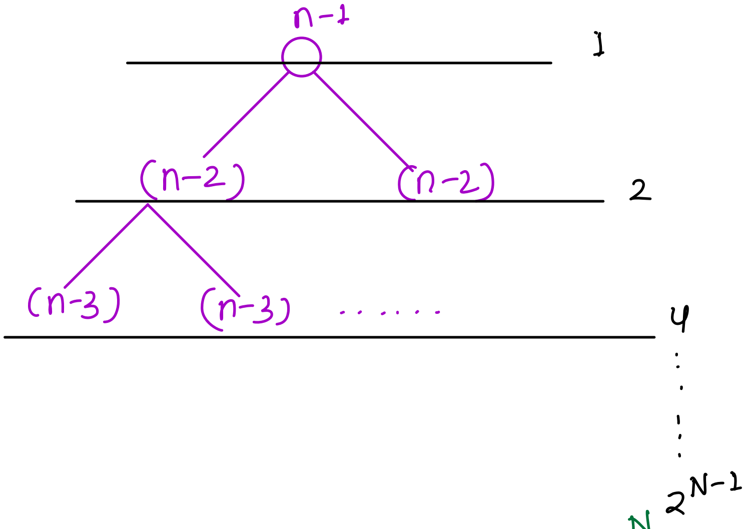
TC : $O(N * 2^N)$

BF 2



```
boolean subsetSum ( index , total ) {
        if ( total == 0 ) return true
        if (index >= N) return false
        take = subsetsum ( index+1 , total - A[index]
        dont = subsetsum ( index+1 , total)
        return take || dont
}
```

TC: $O(2^N)$



Memoization

$\overbrace{0 - N-1}^{N}$ $\overbrace{0 - k}^{k}$

```
HM < String , Boolean > dp
boolean subsetSum (index , total) {
        if ( total == 0 ) return true
        if (index >= N) return false
        if (total < 0) return false
        key = index + '-' + total
        if (dp.containskey(key)) return dp.get(key)
        take = subsetsum ( index+1 , total - A[index])
        dont = subsetsum ( index+1 , total)

        dp.put (key , take || dont)
        return take || dont
}
```

k = target sum

$O(2^N) \longrightarrow O(N * k)$

N = 64 ↓ $\approx 2^{64} \simeq 10^{18}$

k = 100 ↓ 6400

Flipkart's Suggestion Problem

Flipkart wants to make shopping easier for their customers. They plan to ask customers what they need and how much money they want to spend. Then, based on this information, Flipkart will suggest the best products for them to buy. This way, customers can quickly find what they want within their budget and maximizing the customer satisfaction at the same time.

Given budget of user and cost and happiness value of N items of the desired product. Compute max happiness value

Given 0/1

Budget = 300

Namkeen Type	Price	Happiness value
1	110	39
2	180	57
3	50	13
4	120	44
5	100	24

spend = 300

$H = 57 + 44$

$= 101$

Sorting by max Happiness/price will work only if the items can be broken {fractional knapsack}

NOTE ⟶ Always clarify the req.

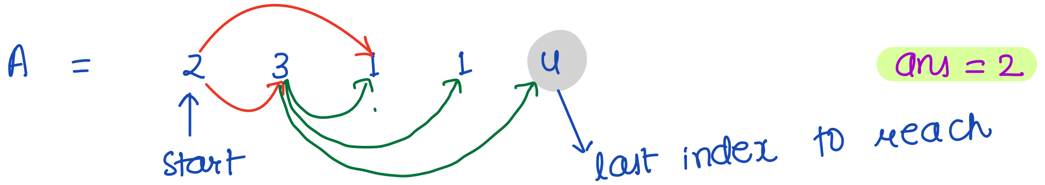
Minimum jumps to reach end *** Amazon

you are given A[N]. you are initially positioned at nums[0].

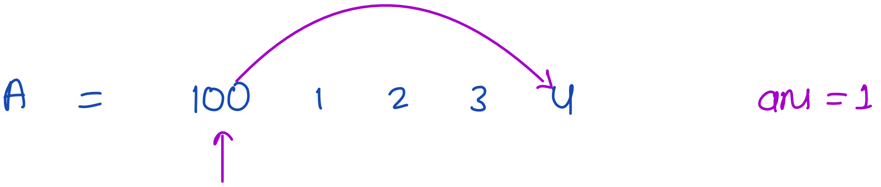
Each A[i] represents the max length of forward jump from index i

Return min no. of jumps to reach nums[n-1]

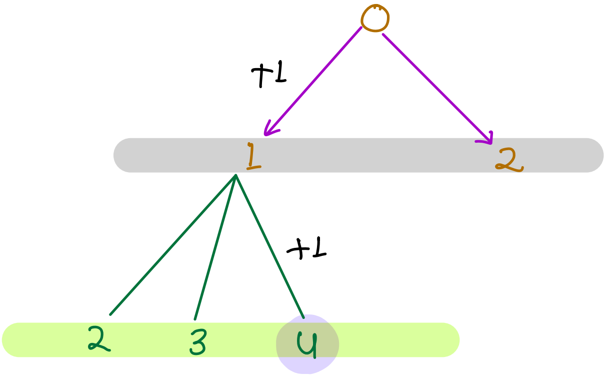
NOTE: you can always reach end.



A = 2 3 0 1 4



0	1	2	3	4
2	3	0	1	4



ans = 2

Pseudocode

```
int minJumps ( index ) {        → 0 to N-1
    if (index >= N-1)  return 0
    jumps = ∞
    // memoise here
    for step → 1 to min(A[index], N)
        nindex = index + step
        jumps = min (jumps , 1+ minJumps ( nindex )
    }
    // here
    return jumps
}
```

TC: $\underbrace{\text{No of unique dp states}}_{N} * \underbrace{\text{TC per state}}_{\max(A)}$

$O(N * \max(A))$

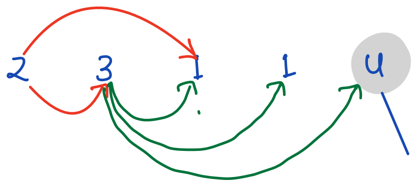
↓

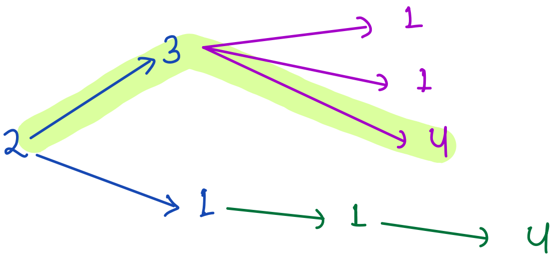
$O(N * N) = O(N^2)$

when does BFS return min no. of steps ?

all edges are same

BFS





Pseudocode

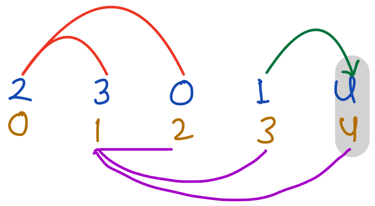
no. of jumps

cur idx

TC: O(N)

```
queue
queue.add({0, 0})
visited = [ ]...false
visited[0] = true

while (!queue.isEmpty()) {
    jumps, idx = queue.remove()
    if(idx >= N-1) { return jumps }
    for step → 1 to min(A[idx], N)
        nidx = idx + step
        if(!visited[nidx]) {
            visited[nidx] = true
            queue.add({jumps+1, nidx})
    }
}
return -1
```



queue ⟶ ~~{0, 0}~~ ~~{1, 1}~~ ~~{1, 2}~~ ~~{2, 2}~~

~~{2, 3}~~ ~~{2, 4}~~ {3, 4}

22 : 34

N digit Numbers

Directi

Find out no. of A digit +ve no. whose digits on being added equals to a given no. B

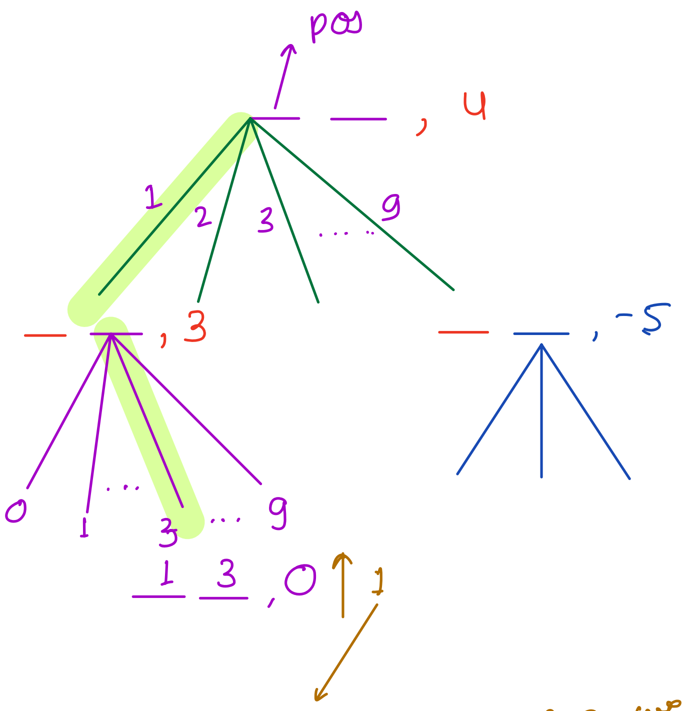
Note : valid no. starts from 1 to 9 except for no. 0 itself ie, leading 0s are not allowed.

output ans % $10^9 + 7$

ans = 4

A = 2 B = 4 → {13 31 22 40}

A = 1 B = 3 → {3} ans = 1



return 1 since we have 1 way

Bruteforce

→ Iterate over all A digit no. $10^{A-1} \ldots \ldots 10^{A}$

check for each number if total == B

TC: $O(A*10^{A})$

Pseudocode

$\overbrace{0}^{A}$ $\overbrace{B}^{B}$ $\overbrace{\text{starting digit } 1}^{2}$

```
int  ndigit ( pos , total , sdigit ) {
    if (total < 0)  return 0  → impossible
    if ( pos == A) {   // exactly A digits
        if (total == 0)  return 1
        return 0          → total exactly 0
    }
    // memoize
    ways = 0

    for digit → sdigit to 9 {
        ways += ndigit( pos+1 , total-digit , 0)
        way %= MOD
    }
    // memoize
    return ways
}
```

TC: O(AB)

Maximum Proofit from stock Prices

DP world

Given an array A where i^{th} element represents the price of stock on day i, the objective is to find the max profit

We are allowed to complete as many transactions as desired but engaging in multiple transactions simultaneously is not allowed. ⟶ Buy before you sell
sell before you buy

$$A = \underset{\text{Buy}}{\boxed{1}} \quad 2 \quad 3 \quad 4 \quad \underset{\text{sell}}{\boxed{5}}$$

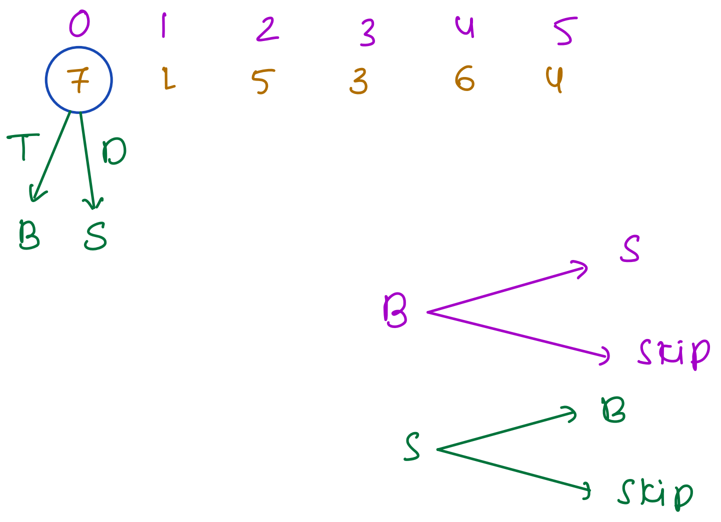
profit = 4

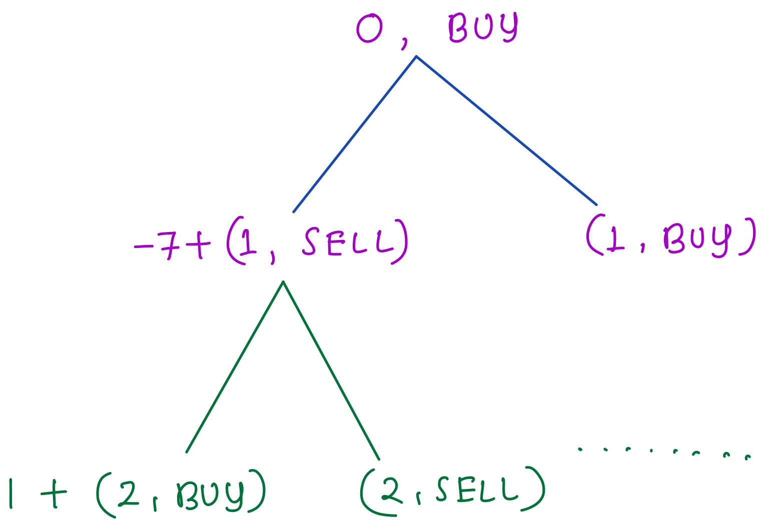
$$A = 7 \quad 6 \quad 4 \quad 3 \quad 1$$

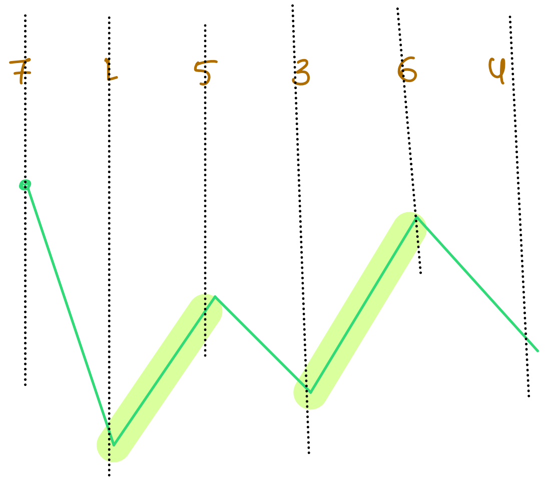
profit = 0

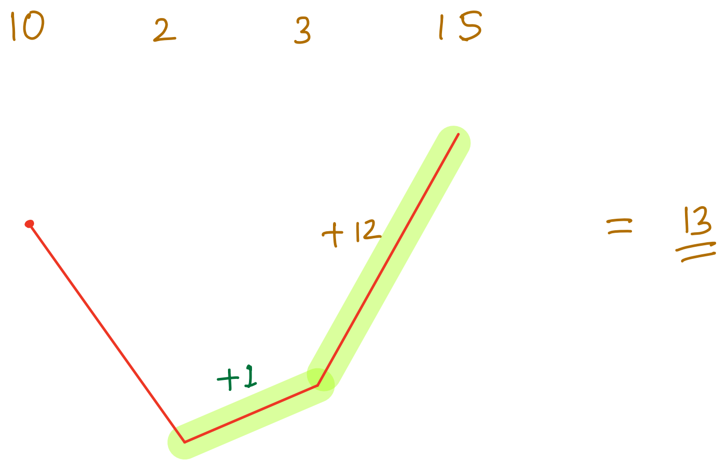
$$A = 7 \quad \overbrace{\underset{B_1}{\boxed{1}} \quad \underset{S_1}{\boxed{5}}}^{4} \quad \overbrace{\underset{B_2}{\boxed{3}} \quad \underset{S_2}{\boxed{6}}}^{3} \quad 4$$

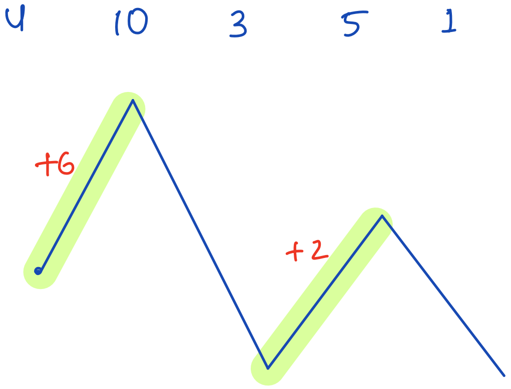
profit = 7




0, BUY
-7+(1, SELL)
(1, BUY)
1 + (2, BUY)
(2, SELL)
....


7
1
5
3
6
4


10
2
3
15
+12
+1
= 13



Pseudocode

```
profit = 0
pprice = A[0]
for (price : A) {
        p = price - pprice
        if (p > 0) profit += p
        pprice = price
}
print(profit)
```