#REVISEWITHARSH #6COMPANIES30DAYS CHALLENGE 2023

CHALLENGE COMPANY 2: Goldman Sachs

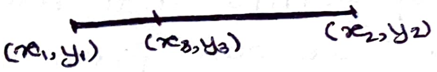
1. *Max Points on a Line*
2. *Valid Square*
3. *Factorial Trailing Zeroes*
4. *Number of Boomerangs*
5. *Split Array into Consecutive Subsequences*
6. *Minimum Consecutive Cards to Pick Up*
7. *Count Good Triplets in an Array*
8. *Maximum Points in an Archery Competition*
9. *IPO*
10. *Number of People Aware of a Secret*
11. *Invalid Transactions*
12. *All Elements in Two Binary Search Trees*
13. *Get Biggest Three Rhombus Sums in a Grid*
14. *Count Nice Pairs in an Array*
15. *Maximum Good People Based on Statements*

leetcode | *LinkedIn* | *GitHub*

Challenge Company 2 : Goldman Sachs

1. Max. Points on a Line

To check the max. no. of points that lie on the same straight line : Simple ans :: SLOPE



→ To ~~find~~ The slope of any two points should be same.

$$\frac{(y_2 - y_1)}{(x_2 - x_1)} = \frac{(y_3 - y_1)}{(x_3 - x_1)} \Rightarrow \boxed{(y_2 - y_1) * (x_3 - x_1) = (y_3 - y_1) * (x_2 - x_1)}$$

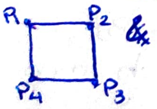
- Make straight line using 2 points & then find how many of the points lie on it.

Time C : $O(n^3)$ Space C : $O(1)$

2. Valid Square

→ A square must have four equal sides, so the set must contain 2 diff. values (length of sides & diagonals).

→ Calculate the distances b/w all pairs of points



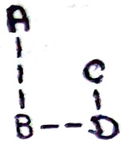
3. Trailing Zeroes

→ To find no. of ~~factorial~~ trailing zeroes in the factorial of no. n, we can find the no. of multiples of 5, 25 & 125 in n, ∵ these are the powers of 5 that will contribute to the no. of trailing zeroes in the factorial.

→ To find the no. of multiples of 5, we n/5. This will give us the no. of multiples of 5 in range 1-n ∴ the no. of multiples of 5 in the factorial of n.

→ To find total no. of trailing zeroes in the factorial of n, we add up the no. of multiples of 5, 25 & 125 in the total no. of trailing zeroes in factorial. factorial. This will give us?

4. Number of Boomerangs, to calculate it, we need to consider each point in a plane & count the no. of pairs of points that have the same distance from the current point. for eg., Here, point A is the current point, & point B & C are two pairs of points that have same distance from A. ∴ we can form two boomerangs using A, B & C.



5. Split Array into Consequtive Subsequences

e.g., nums = ~~[1,2,3,4,5,6,7]~~ [1,2,3,4,4,5] Output = True

- The method first creates a 'countMap' with the frequency of each no. in input array which would be : {1:1, 2:1, 3:1, 4:2, 5:1}
- Then it iterates through the input array, it will pick each no. & check if it can form a sequence i.e., atleast of length 3, & composed of consequtive numbers
- It will see that first subsequence is [1,2,3], the second subsequence is [4,4,5] which are both are atleast 3 in length & consecutive. So it will return True.

e.g., nums = [1,2,3,4,4] Output = False

- In this case, it will see that only two subsequence can be formed which is [1,2,3],[4,4] which are not atleast 3 in length. So it will return False

6. Min. Consequtive Cards to Pick up.

e.g., Input : cards = [1,2,3,4,4,4,4,4,4]; Output = 8 2
The duplicate card 4 is at index 4,5,6,7,8.
To pick all duplicate cards, you need to pick up the cards from index 4 to 8 inclusive, which is a total of 8 cards.

→ In this e.g., the TreeMap will contain {1 = (1,0), 2 = (1,1), 3 = (1,2), 4 = (6,4)} & the final ans will be 8.

- when the first duplicate card 4 is found at index 4, it updates ans to be the current no of cards picked up, which is 1 (the card at index 4).
- when next duplicate card 4 is found at index 5, ans = 2.
- ∵ all other duplicate cards are also at indices 6, 7 & 8, the no. of cards picked up remains 2. ∵ all cards are already picked up.
- ∴ ans was not updated any further, the final output is 2, as it is the min. no. of cards that must be picked up in order to collect all the duplicate cards.

8. Max. Points in an Archery Competition.

e.g., Bob has 8 arrows & Alice has [6, 10, 2] arrows in each round. The func^n first tries using 0 arrows in each round, but that gives Bob 0 points. Then it tries using 1 arrow in round 1, 0 in 2 & 0 in 3, but that gives Bob only 1 point. & so on, it tries all possible combination of arrows.
In the end, it finds that the best strategy ...

for Bob is using 2 arrows in round 1,6 in 2 & 0 in 3, thus giving Bob total 6 points, which is the highest possible score that can be obtained with 8 arrows. This strategy is saved in 'optimalStrategy' array i.e., [2,6,0]

9. IPO

e.g., Initial capital = 0 & the company can invest in 2 projects. The profits & capital requirements of the projects are [1,2,3] & [0,1,1] resp. The method would sort the project by capital req., which would give the order [0,1,1],[1,2,3]. The priority queue would ~~start~~ off empty, & the company would choose the first project, with the profit of ~~one~~ 1. & capital req of 0, increasing its capital to 1. Then the 2nd project would be added to the priority queue with the profit of 2 & the capital req. of 1. The company can then invest in this second project, increasing its capital to 3. ∵ no more investments are available, the method returns 3 as the final capital.

10. No. of People Aware of a Secret

e.g., The No. of days = 5, the delay = 2 & forget = 3.

- Initially, only 1 person knows the secret.
- On 2nd day, 1 more person knows the secret, so totalAwarePeople = 2
- On 3rd day, 1 more person knows the secret, so totalAwarePeople = 3
- On 5th day, 1 more person knows the secret, so totalAwarePeople = 4.
- At the end it will remove the last forgotten people, so totalAwarePeople = 3

11. Invalid Transactions

e.g., If the input transactions are:
["alice, 20, 1200, ny", "bob, 10, 700, la", "alice, 20, 900, ny"],
this code will mark the first transaction as invalid (amount > 1000) & the list of invalid transactions returned would be ["alice, 20, 1200, ny"]

12. All Elements in 2 BST

e.g., root1: [2, 1, 4] and root2 = [1, 0, 3]
Output: [2, 1, 1, 2, 3, 4]

- It will take values from both trees using pre-order traversal [2, 1, 4] & [1, 0, 3]
- will append it to the list: [2, 1, 4, 1, 0, 3]
- then it will sort the list: [0, 1, 1, 2, 3, 4]
- return the final list [0, 1, 1, 2, 3, 4]

15. Max. Good People Based on Statements

e.g., if given input of 2D Array is

1 1 0
1 0 0
0 0 0

The max. no. of good people is 2
(people at index 0 & 1 are good).