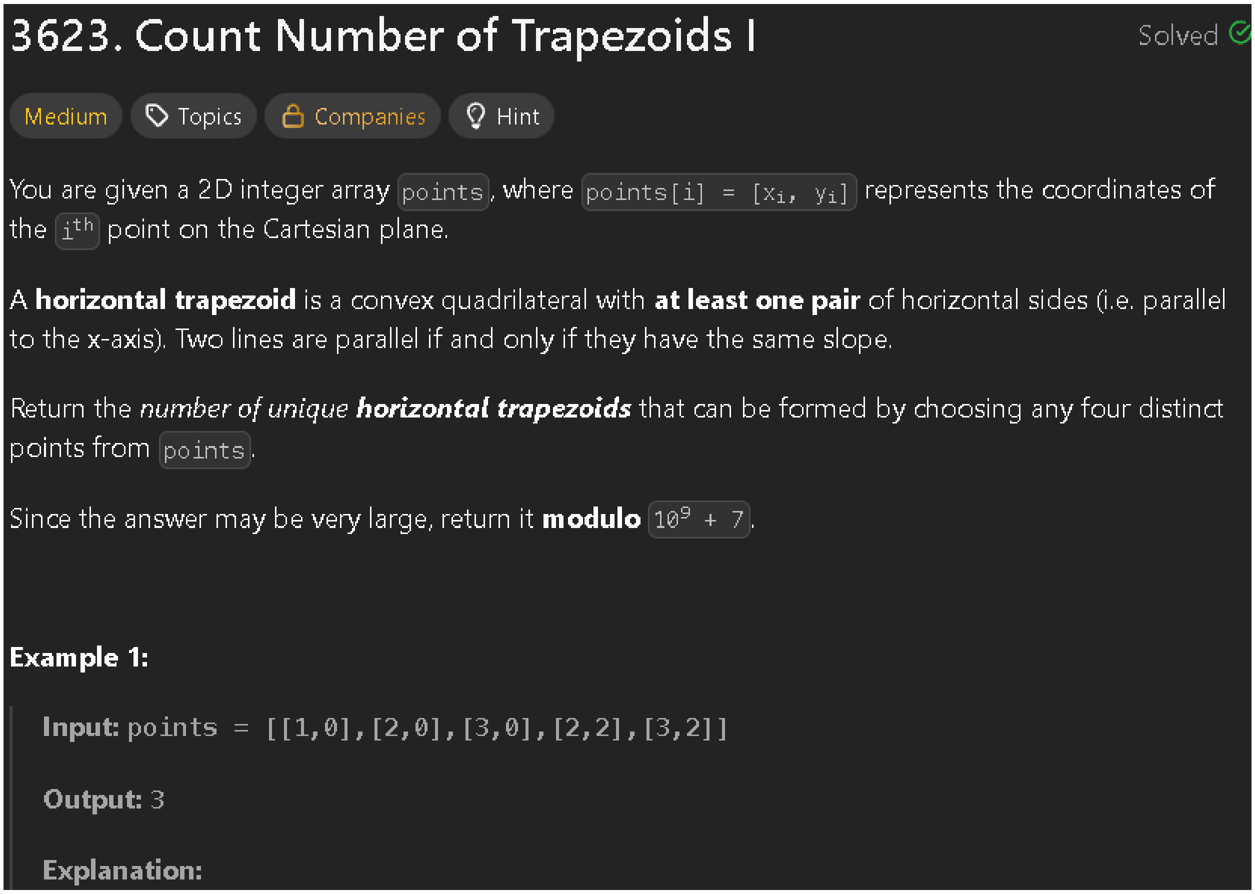


3623. Count Number of Trapezoids I

You are given a 2D integer array `points`, where `points[i] = [xi, yi]` represents the coordinates of the `ith` point on the Cartesian plane.

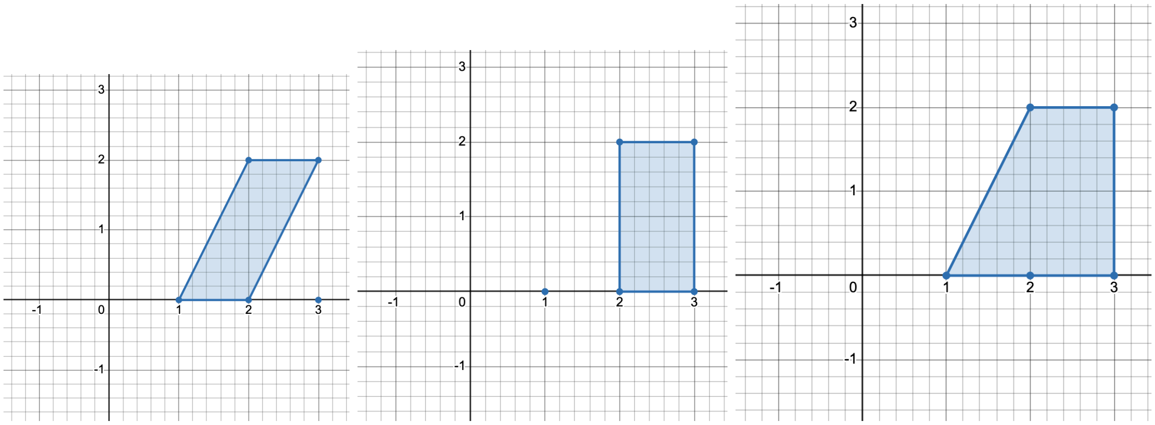
A **horizontal trapezoid** is a convex quadrilateral with **at least one pair** of horizontal sides (i.e. parallel to the x-axis). Two lines are parallel if and only if they have the same slope.

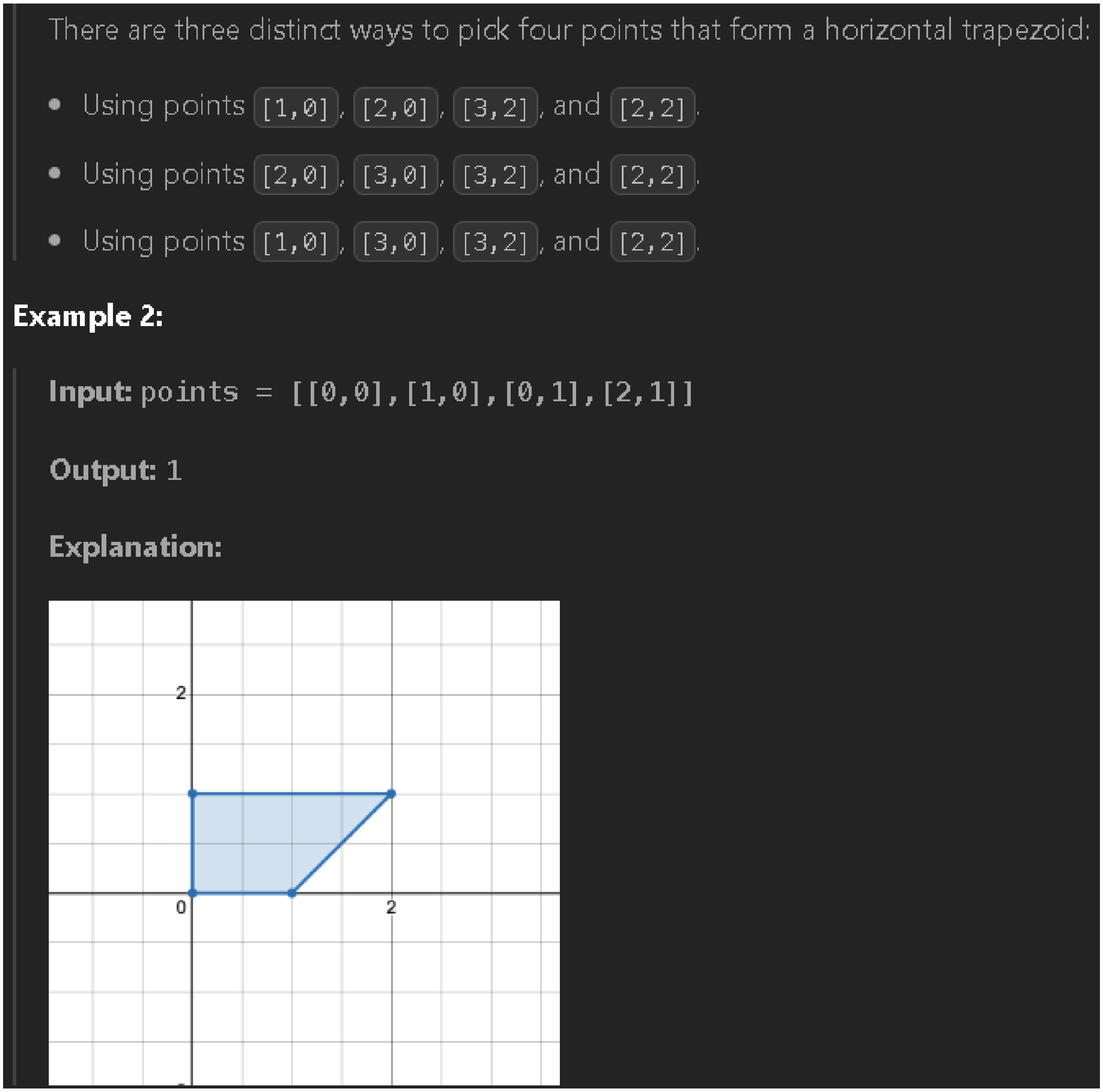
Return the *number of unique* ***horizontal trapezoids*** that can be formed by choosing any four distinct points from `points`.

Since the answer may be very large, return it **modulo** $10^9 + 7$.

Example 1:

> **Input:** points = [[1,0],[2,0],[3,0],[2,2],[3,2]]
>
> **Output:** 3
>
> **Explanation:**

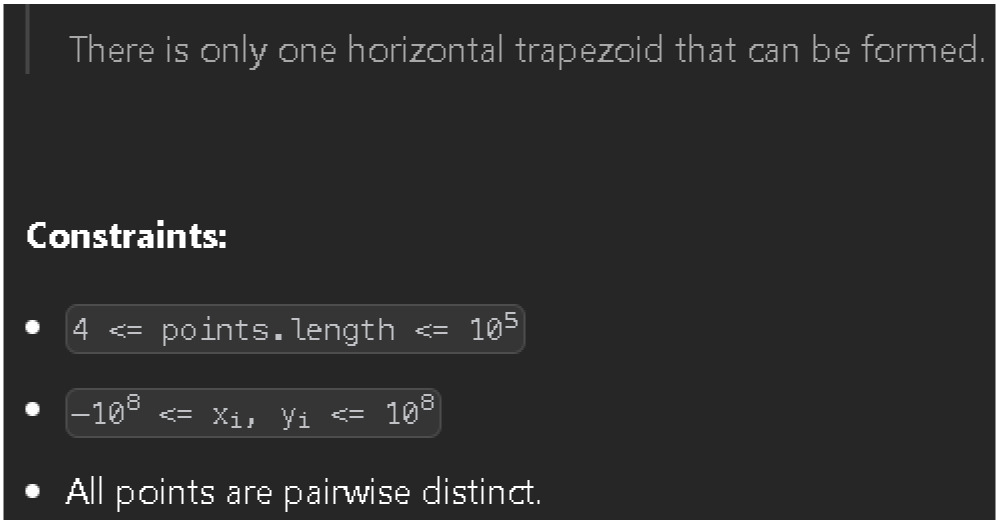




> There are three distinct ways to pick four points that form a horizontal trapezoid:
>
> - Using points `[1,0]`, `[2,0]`, `[3,2]`, and `[2,2]`.
> - Using points `[2,0]`, `[3,0]`, `[3,2]`, and `[2,2]`.
> - Using points `[1,0]`, `[3,0]`, `[3,2]`, and `[2,2]`.

Example 2:

> **Input:** `points = [[0,0],[1,0],[0,1],[2,1]]`
>
> **Output:** 1
>
> **Explanation:**



> There is only one horizontal trapezoid that can be formed.

Constraints:

- $4 <= points.length <= 10^5$
- $-10^8 <= x_i, y_i <= 10^8$
- All points are pairwise distinct.

Python:

```
class Solution:
    def countTrapezoids(self, points: List[List[int]]) -> int:
        MOD = 1000000007
        groups = defaultdict(int)
        for _, y in points:
            groups[y] += 1
        res = total = 0
        for y, count in groups.items():
            lines = count * (count - 1) // 2
            res = (res + total * lines) % MOD
            total = (total + lines) % MOD
        return res
```

JavaScript:

```
/**
 * @param {number[][]} points
 * @return {number}
 */
var countTrapezoids = function(points) {
    const MOD = 1000000007n;
    const cnt = new Map();
    for (const p of points) {
        const y = p[1];
        cnt.set(y, (cnt.get(y) || 0n) + 1n);
    }
    const seg = [];
    for (const k of cnt.values()) {
        if (k >= 2n) seg.push(k * (k - 1n) / 2n);
    }
    let sum = 0n, ans = 0n;
    for (const v of seg) {
        ans = (ans + v * sum) % MOD;
        sum = (sum + v) % MOD;
    }
    return Number(ans);
};
```

Java:

```
class Solution {
    public int countTrapezoids(int[][] points) {
        int n = points.length, MOD = 1000000007;
        Arrays.sort(points, Comparator.comparingInt(a -> a[1]));
        long res = 0, total = 0;
        for (int i = 0, j; i < n; i = j){
            j = i + 1;
            while (j < n && points[i][1] == points[j][1]){
                j++;
            }
            long count = j - i;
            long lines = count * (count - 1) / 2;
            res = (res + total * lines) % MOD;
            total = (total + lines) % MOD;
        }
        return (int)res;
    }
}
```