**2251. Number of Flowers in Full Bloom: -**

**Hard Accepted: 39K Submissions: 71.8K Acceptance Rate: 54.4%**

You are given a **0-indexed** 2D integer array flowers, where flowers[i] = [starti, endi] means the ith flower will be in **full bloom** from starti to endi (**inclusive**). You are also given a **0-indexed** integer array people of size n, where people[i] is the time that the ith person will arrive to see the flowers.

Return *an integer array*answer*of size*n*, where*answer[i]*is the****number****of flowers that are in full bloom when the*ith*person arrives.*

**Example 1:**

A diagram of a number of objects

Description automatically generated

**Input:** flowers = [[1,6],[3,7],[9,12],[4,13]], poeple = [2,3,7,11]

**Output:** [1,2,2,2]

**Explanation:** The figure above shows the times when the flowers are in full bloom and when the people arrive.

For each person, we return the number of flowers in full bloom during their arrival.

**Example 2:**

A number of flowers with numbers

Description automatically generated with medium confidence

**Input:** flowers = [[1,10],[3,3]], poeple = [3,3,2]

**Output:** [2,2,1]

**Explanation:** The figure above shows the times when the flowers are in full bloom and when the people arrive.

For each person, we return the number of flowers in full bloom during their arrival.

**Constraints:**

* 1 <= flowers.length <= 5 \* 104
* flowers[i].length == 2
* 1 <= starti <= endi <= 109
* 1 <= people.length <= 5 \* 104
* 1 <= people[i] <= 109

**Code: -**

class Solution {

public:

    vector<int> fullBloomFlowers(vector<vector<int>>& flowers, vector<int>& people) {

        map<int,int> mp;

        mp[0] = 0;

        for(auto &v : flowers){

            mp[v[0]] += 1;

            mp[ v[1]+1 ] -= 1;

        }

        int prev = 0;

        for (auto &p : mp){

            p.second = prev + p.second;

            prev = p.second;

        }

        int n = people.size();

        vector<int> ans(n);

        for(int i = 0; i<n; ++i){

            if(mp.find(people[i]) != mp.end())

                ans[i] = mp[ people[i] ];

            else{

                auto it = mp.lower\_bound( people[i] );

                --it;

                ans[i] = (\*it).second;

            }

        }

        return ans;

    }

};

**T.C: - O(N \* log N)**

**S.C: - O(N)**