Question :

Problem.-> Given an array of size “N”; find the number of triplets; such that A[i] >A[j]< A[k] such that i < j < k ;

1<=N<=1000

Brute Force :

* Exploring all possibilities and the TC wil be O(N^ 3)

Step 1 : For the given constrain that , a[i] > a[j] and a[j] < a[i]

Step 2 : For each index , there can be a possible triplet for this , so we need to calculate what are the numbers which are less than[j] the current index [i].

Step 3 : We can store it as prefix array

Step 4 : For each index , we need to know what are the numbers which are less than [j] the current index [k] store it in it suffix.

Step 5 : So , for index I , we know what er the numbers which are less than the current index and graeter than the current index .

Step 6 : for index I , multiply prefix[i] & suffix[i]

class Solution {

    public int maxTriplets(int [] nums) {

        int n = nums.length;

        int [] prefix = new int[n];

        int [] suffix = new int[n];

        prefix[0] = 0;

        suffix[n - 1] = 0;

        for(int i = 1 ; i < n ; i++){

            int cnt = 0;

            for(int j = i - 1 ; j >= 0 ; j--){

                if(nums[i] > nums[j]){

                    cnt++;

                }

            }

            prefix[i] = cnt;

        }

        for(int i = n- 1 ; i >= 0 ; i--){

            int cnt = 0;

            for(int j = i + 1 ; j < n; i++){

                if(nums[i] < nums[j]){

                    cnt++;

                }

            }

            suffix[i] = cnt;

        }

        int ans = 0;

        for(int i = 0 ; i < n ; i++){

            ans += prefix[i] \* suffix[i];

        }

    }

}