# Contest Problem C - 100300. Sum of Digit Differences of All Pairs

You are given an array nums consisting of positive integers where all integers have the same number of digits.
The digit difference between two integers is the count of different digits that are in the same position in the two integers.
Return the sum of the digit differences between all pairs of integers in nums.

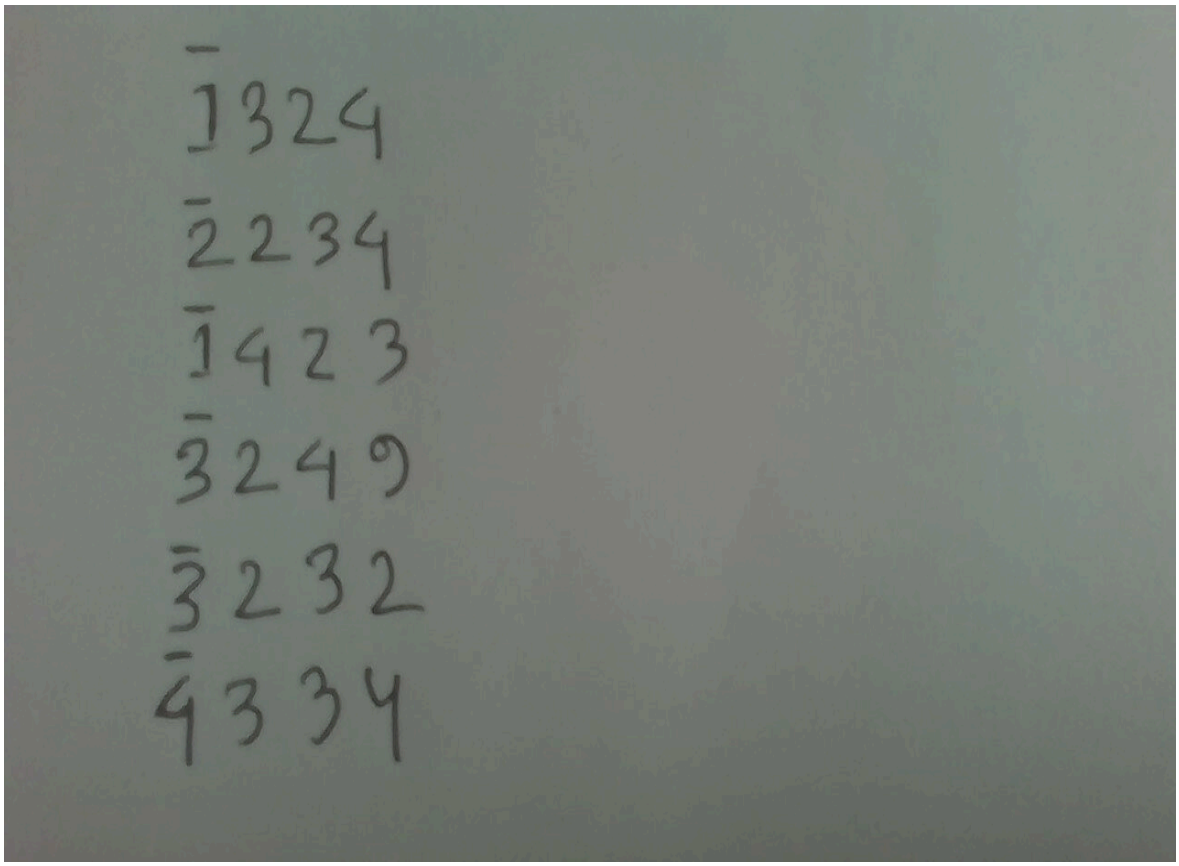
Example 1:
Input: nums = [13,23,12]
Output: 4
Explanation:
We have the following:
- The digit difference between 13 and 23 is 1.
- The digit difference between 13 and 12 is 1.
- The digit difference between 23 and 12 is 2.
So the total sum of digit differences between all pairs of integers is 1 + 1 + 2 = 4.

Example 2:
Input: nums = [10,10,10,10]
Output: 0
Explanation:
All the integers in the array are the same. So the total sum of digit differences between all pairs of integers will be 0.

Constraints:
2 <= nums.length <= 10^5
1 <= nums[i] < 10^9
All integers in nums have the same number of digits.

## **Intuition**:

1. First observation is if i work with digits it gets easier, all i have to deal is maximum 8 digits as nums[i]<10^9;
2. The Contribution trick comes handy here, for every digit position what is the maximum contribution of that digit position?
if there are total n numbers and all of them are different in the particular digit position, it contributes (n*(n-1))/2.



For example in the photo the first digit position maximum contribution is ( 6 * 5 )/2 = 15
But there are two duplicates 1 twice and 3 twice. Two 1's make pair with itself and negative contribute 1 as (1,1) has no digit difference. So as two 3's.
The first place digit position will contribute 15-1-1 = 13

So the solution is to calculate the summation of contribution for every digit position.

```
class Solution {
public:
    long long sumDigitDifferences(vector<int>& nums) {
        long long int ans = 0;
        long long int n = nums.size();
        vector<vector<int>> arr;
        for(int i=0; i<n; i++){
            int a = nums[i];
            vector<int> vec;
            while(a){
                int x = a%10;
                a = a/10;
                vec.push_back(x);
            }
            arr.push_back(vec);
        }
        int s = arr[0].size();
        for(int i=0; i<s; i++){
            map<int,long long int> mp;
            long long int pvt = (n*(n-1))/2;
            for(int j=0; j<n; j++){
                mp[arr[j][i]]++;
            }
            for(auto it:mp){
                pvt -= (it.second*(it.second-1))/2;
            }
            cout<<pvt<<"\n";
            ans+=pvt;
        }
        return ans;
    }
};
```