Qn Link : <https://www.desiqna.in/16227/uber-oa-sde-1-ctc-65-l-jan-2024>

Question Summary:

* You are given an array of size “N” and find the sum of all possible pairs .
* Such that func(x , y) = xy + yx.

Brute force :

class Solution {

    public int solution(int [] nums , int X , int Y) {

        int n =  nums.length;

        int sum = 0

       for(int i = 0 ; i < n ; i++){

        for(int j = 0 ; j< n ; j++){

            String a = Integer.toString(nums[i]);

            String b = Integer.toString(nums[i]);

            String c = a + b;

            sum += Integer.parseInt(c);

        }

       }

       return sum ;

    }

}

Optimized Approch :

Consider the given array [2 , 1 , 3]

* The possible pairs of sum are 22 , 21 , 23 , 12 , 11 , 13 , 32 , 31 , 33

So , now how can we generate these pairs in O(N ) time ??

{15 , x , y , z } 🡪 1515 , 15x , 15y , 15z these are the possible pairs for 15 , 🡪 x15 , xy , xz ……..

{15 , 1 , 2 , 10}

🡪1515 , 151 , 152 , 1510 🡪 1500 + 150 + 150 + 1500 + (sum of array)

🡪115 ,11 , 12 , 110 🡪 (100 + 10 + 10 + 100) + (sum of array)

🡪215 , 21 , 22 , 210 🡪 (200 + 20 + 20 + 200) + (sum of array)

🡪1015 , 101 , 102 , 1010 🡪 (1000 + 100 + 100 + 1000) + (sum of array)

* So basically the RHS is sum of the array , now the only concern is LHS.
* The sum (1.5k + 150 + 150 + 1.5K) are basically the power of length of numbers
* So we can calulate how many times length of each digit appears \* current number \* 10 ^ l

Atlast , the right sum

class Solution {

    public int solution(int [] nums , int X , int Y) {

        int n =  nums.length;

        long sum = 0;

        int [] digits = new int [10];

        //Counting the length of each digit

        for(int i = 0 ; i < n ; i++){

            digits[new String(nums[i])]++;

        }

        for(int i = 0 ; i< n ; i++){

            int length = 1;

            //length of number varies from 1 to 6 because the constains is 10 ^ 5

            while(length <= 6){

                // calculating the sum at each index (LHS)

                // sum + nums[i] \* length of the number \* occurence of the number with length L

                sum = sum + (nums[i] \* Math.pow(10 , length) \* digits[length]);

                length++;

            }

            int right = sum of Array;

            int left = sum;

             ans += left + right;

        }

        return ans;

    }

}