max Arithmetic Length
input A:[0, 4, 8, 16]
input B:[0, 2, 6, 12, 14, 20]
output: 新的A 的长度
从B中最多能选几个数字使得 A 变成一个任意两个数之间的差是相等的数组， 例如：[0, 4, 8, 12, 16, 20]
千年难题，还是不会

---

1. GoodTuple

链接里第一题：Here
Give an array and find the count of a pair number and a single number combination in a row of this array. Target array is a[i – 1], a, a[i + 1]
Example：
Input: a = [1, 1, 2, 1, 5, 3, 2, 3]
Output: 3
Explain:
[1, 1, 2] –> two 1 and one 2(O)
[1, 2, 1] –> two 1 and one 2(O)
[2, 1, 5] –> one 2, one 1 and one five(X)
[1, 5, 3] –> (X)
[5, 3, 2] –> (X)
[3, 2, 3] –> (O)

```
        int result = 0;
        if (nums.length <= 2) {
            return 0;
        }
        int pre1 = nums[1];
        int pre2 = nums[0];
        for (int i = 2; i < nums.length; i++) {
            if (nums[i] != pre1 && nums[i] != pre2 &&  pre1 != pre2) {
                result++;
            }
            pre2 = pre1;
            pre1 = nums[i];
        }
```

---

2. SumOfString

链接里第二题：Here
String fun(String a, String b) a 和 b数字组成, a和b的第ith个数字依次相加, 返回新String
 eg. a = "99" b = "1"   return   "910"
如果写Java的话最好用StringBuilder, String 会 TLE

---

3. Divide Array into two with equal size, make sure every element in each two array is unique. return empty list if impossible.
先sort再分配，连续超过2，就不能满足了
我是一个HashMap记录原数组每个数出现次数，如果有某个超过2的话，就不可能满足要求直接返回空；如果出现次数都是2或以下的话，就对出现两次的平分到两个数组，剩下只出现一次的随便分给两个数组就行只要保障最后两个数组长度一致。

---

4. Cool Feature:
链接里第四题：Here
最开始有3个case超时，原因是每次执行问加起来等于target的情况有多少种的那个query时，我都是重新算一遍记录b数组每个数出现次数的HashMap，其实只用最开始构好，然后每次做第二种query([index, num])，更新一下HashMap就行。改了就过了。
Give three array a, b and query. This one is hard to explain. Just read the example.

Input:
a = [1, 2, 3]
b = [3, 4]
query = [[1, 5], [1, 1, 1], [1, 5]]
Output:
[2, 1]
Explain:
Just ignore every first element in sub-array in the query.
So we will get a new query like this query = [[5], [1, 1], [5]]
Only record the result when meet the single number in new query array.
And the rule of record is find the sum of the single number.
The example above is 5 = 1 + 4 and 5 = 2 + 3, there are two result.
So currently the output is [2]
When we meet the array length is larger than 1, such as [1, 1]. That means we will replace the b[x] = y, x is the first element, y is second element. So in this example, the b will be modify like this b = [1, 4]
And finally, we meet the [5] again. So we will find sum again. This time the result is 5 = 1 + 4.
So currently the output is [2, 1]
note: Don't have to modify the query array, just ignore the first element.
Time:
Function findSum is O(a * b)
Function modifyArrayb is O(1)
Function treverse is O(query)
So total maybe O(a * b * query)
I think this problem must has some better solution, but I am almost run out of time.

```
public List<Integer> coolFeature(int[] a, int[] b, int[][] query){
        Map<Integer, Integer> mapA = new HashMap<>();
        for(int num: a)
            mapA.put(num, 1 + mapA.getOrDefault(num, 0));
        List<Integer> res = new ArrayList<>();
        for(int[] q: query){
            // modify:
            if(q.length == 3){
                b[q[1] - 0] = q[2];
            }
            // find sum, quite like 2-sum problem
            else{
                int sum = q[1], count = 0;
                for(int i = 0; i < b.length; ++i){
                    int target = sum - b[i];
                    if(!mapA.containsKey(target)) continue;
                    count += mapA.get(target);
                }
                res.add(count);
            }
        }
        return res;
    }
```

---

5. most Frequent Digits
出现频率最高的数字
Input: A = [22, 2, 3, 33, 5]
Output: [2, 3]
开始有一个case过不了，后来重新初始化了一下dictionary， 可能是A=[]的情况

---

6. rotate Over Diagonals
这个题大家都没说是转K次 但是 diag不转 搞得我第一次运行发现有bug才改过来
Example:
[[1, 2, 3],
 [4, 5, 6],
 [7, 8, 9]]
-->
[[1, 4, 3],
 [8, 5, 2],
 [7, 6, 9]]

[[1, 2, 3, 4, 5],
 [6, 7, 8, 9, 10],
 [11,12,13,14,15],
 [16,17,18,19,20],
 [21,22,23,24,25]]
-->
[[1,16,11,6,5],
[22,7,12,9,2],

[23,18,13,8,3],
[24,17,14,19,4],
[21,10,15,20,25]]

7. matrixQueries
给一个array和一个matrix。
matrix里面每一个vector<int>的形式必定是[l,r,target]，固定只有3个数。
然后要求统计array里 index从l 到 r这个区间出现了多少次target这个数。
比如：
array = [1,1,2,3,2]
matrix = [[1,2,1],
              [2,4,2],
              [0,3,1]]
output : 5

因为在matrix[0]，array的index 1到2区间出现了1 一次，
matrix[1], array的index 2到4区间出现2 两次。
matrx[2], array的index 0到3区间出现1 两次

这个题如果直接暴力解O(n*n)会有两个test case过不了。我是用hashmap<int, vector<pair<int,int>>>。 key是target， value是index区间。
这样走一遍array，每次确定一下当前index在不在区间里就行了。
1， [[1,2],[0,1]]
2，[[2,4]]

然后loop一遍array,
i =0, arr *= 1,   然后这个时候判断map.containKey(arr)，然后走一遍key里的value，因为 0 <= i <= 1，所以output++;*

8. 花式数位求和
25631 –> 2 – 5 + 6 – 3 +1

9. 查询矩阵
给int n, m，想象n*m的矩阵M，M[i,j] = (i+1)*(j+1)，0–based
一系列query，有三种类型，第一种是查询矩阵中最小的元素，第二、三分别是禁用某一行、列。
一个2D array的min number的query
题目是说给你一个2d array。其中array*[j] = (i+1)*(j+1)。这个给定。*
然后给一堆query，有三种不同的格式：
第一种是让你返回当前array中的最小值
第二种是让你把某一行disable
第三种是把某一列disable
当然disable了之后最小值就不能用了

10.Keyboard: input: String, char[] array; output int返回可以输出的字符串个数

这道题我看之前各位的面经一直对什么地方该split什么地方不该感到困惑，结果还是让我遇到了，仔细看题之后发现只需要按照空格分就行了，
broken keyboard 键盘的部分英文字母键坏了（注意只有字母键坏了） 给定一个String 和 一个char Array（没坏的字母键），输出String中能打出的字符串数。
栗子：
input “hello, world!” ['i','e','o','l','h']; output: 1 (只能打出 hello 这个单词)
input “5 + 3 = 8” []; output: 5 (没有英文字母， 5， +， 3， =， 8 都可以打出)
之前面经有过的题。输入一组words和一组valid letters，判断有多少个words是valid。判断条件是words里的所有upper and lower letter必须在valid letters里面。如果word里面有special character不用管。注意valid letter只有小写，但是words里面有大写的也算valid。比如words = [hEllo##, This^^], valid letter = [h, e, l, 0, t, h, s]; "hello##" 就是valid，因为h，e，l，o都在valid letter 里面，“This^^” 不valid, 因为i不在valid letter里面

11 给一个数num, 返回这个数每一个digit的乘积: product 和 每一位digit的和: sum的差值

12 反转连续的两个字符, "abcdef" => "badcfe"

13 找到出现频率最多的数字, 升序输出, [1,1,2,3,4,4] => [1,4]

14 **divisorSubstrings** 给一个数字, 和一个 k 值(表示除数位数), 看能用多少个sub number整除
ex: n = 1220 k = 2 => 1220 % 12 != 0, 1220 % 22 != 0, 1220 % 20 == 0 => ans : 1

15 **sort diagnal** https://www.1point3acres.com/bbs/thread-546408-1-1.html 第二题
给2D array, 斜的(方向从左上到右下)为一列做sorted
ex :
4, 5, **3** 　　**2**, 1, **3**
3, **2**, 1 　=> 　1, **3**, 5
2, 1, **3** 　　**2**, 3, **4**

ex:
1, 3, **9**, 4 　　**1**, 3, **7**, 4
9, **5**, 7, **7** 　=> 　2, **2**, 7, **9**
3, 6, **9**, 7 　　**2**, 6, 5, 7
1, **2**, 2, **2** 　　1, **3**, 9, **9**

Question 3: divisorSubStrings

16. 之前面经也出现过。compare两个string，只有小写字母。 每个stirng内部可以任意换位置，所以位置不重要。每个string内部两个letter出现的频率也可以互换，所以这题只需要两个string每个frequency出现的次数要一样。
比如“babzccc” 和 “bbazzcz” 就返回“true”，因为z和c可以互换频率。 但是“babzcccm” 和 “bbazzczl” 就不一样，因为m在第一个里出现过，第二个里没有出现过。
Given two rules to define two strings are close enough.
1. you can swap neighbor char any times. Ex. "abb" -> "bba"
2. If two strings have the same character, then you can change the character into another.

Ex. If both strings contain "a" and "b", you can change all "a"s in the first string or change all "b"s in the first string. same as the second string

Ex.
Input: S1 = "babzccc", S2 = "abbzczz"
Output: True

Sol.
Use a dictionary to record the frequency of characters.
Remove the same part in dictionaries
try to find the pair that have different character but with the same frequency

17 4. Maximum size of ribbon

Given a list representing the length of ribbon, and the target number "k" parts of ribbon.
we want to cut ribbon into k parts with the same size, at the same time we want the maximum size.

Ex.
Input: A = [1, 2, 3, 4, 9], k = 5
Output: 3

Explanation:
if size = 1, then we have 19 parts
if seize = 2, then we have 8 parts
if size = 3, then we have 5 parts
if size = 4, then we have 3 parts, which is not enough.
So return the max size = 3.

Sol.
Use binary search to find the size of ribbon to reach the time limit.
if size = 1, [1//1, 2//1, 3//1, 4//1, 9//1] -> sum([1, 2, 3, 4, 9]) = 19 parts
if seize = 2, [1//2, 2//2, 3//2, 4//2, 9//2] -> sum([0, 1, 1, 2, 4]) = 8 parts

https://leetcode.com/discuss/interview-question/349634/Quora-or-Online-Assessment-with-CodeSignal

https://docs.google.com/document/d/1lt5Fd516j9ch6t2Yn1ylh1wlYSxyDurTp2Lmvo-4aR4/edit