**67. Add Binary**

Given two binary strings a and b, return *their sum as a binary string*.

Input: a = "1010", b = "1011"

Output: "10101"

**844. Backspace String Compare**

Given two strings s and t, return true *if they are equal when both are typed into empty text editors*. '#' means a backspace character.

Note that after backspacing an empty text, the text will continue empty.

**Input:** s = "ab#c", t = "ad#c"

**Output:** true

**Explanation:** Both s and t become "ac".

**Input:** s = "a#c", t = "b"

**Output:** false

**Explanation:** s becomes "c" while t becomes "b".

**1002. Find Common Characters**

Given a string array words, return *an array of all characters that show up in all strings within the*words*(including duplicates)*. You may return the answer in **any order**.

**Input:** words = ["bella","label","roller"]

**Output:** ["e","l","l"]

**Sol:**

Initialize the array with max value

int[] minFrequencies = new int[26];

Frequency for bella : [1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Once we get the char count we find out

minFrequencies[i] = Math.*min*(minFrequencies[i], charFrequencies[i]);

Frequency for label: [1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

minFrequencies[i] = Math.*min*(minFrequencies[i], charFrequencies[i]);

Frequency for roller: [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 0, 0, 1, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0]

minFrequencies[i] = Math.*min*(minFrequencies[i], charFrequencies[i]);

in last we will get

common char count = [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

from this count we will find out the chars.

<https://github.com/hareramcse/Datastructure/blob/master/String/src/com/hs/leetcode/CommonCharFromNString.java>

**409. Longest Palindrome**

Given a string s which consists of lowercase or uppercase letters, return *the length of the****longest palindrome*** that can be built with those letters.

Letters are **case sensitive**, for example, "Aa" is not considered a palindrome here.

**Input:** s = "abccccdd"

**Output:** 7

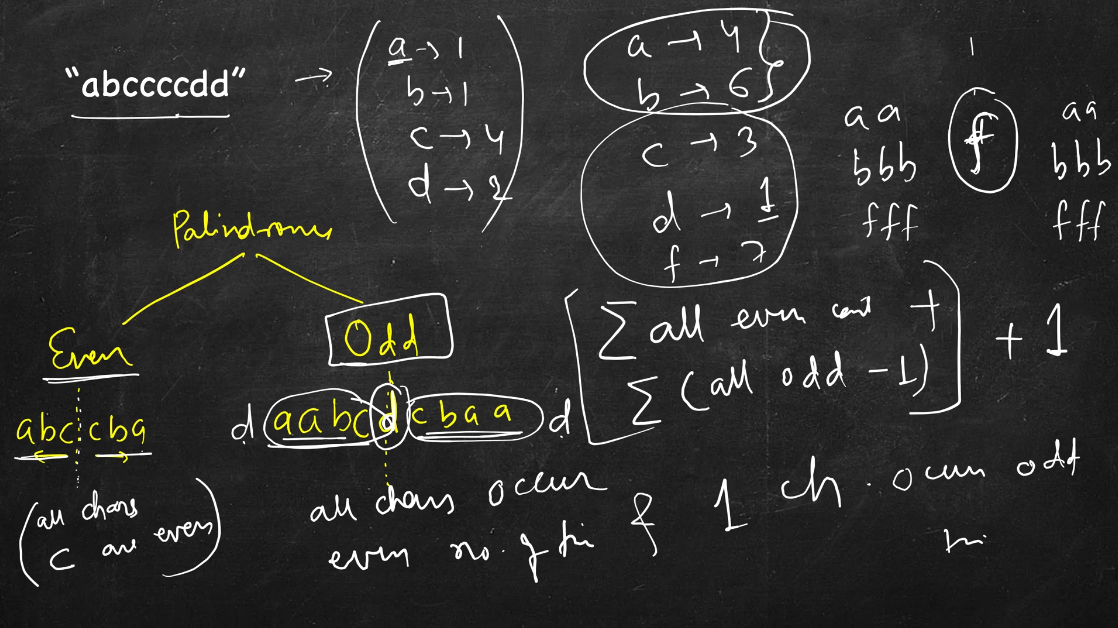
**Explanation:** One longest palindrome that can be built is "dccaccd", whose length is 7.

**Sol:**

Take the map and store the count of each char in of the input.

Palindrome can be two type: even length palindrome and odd length palindrome.

If the count of the char is even then we can simply keep same number of char to both side. If it is odd then we will make this char as middle element of the palindrome and reduce its size by 1 so that it will be even length and then we will equally put in both half.



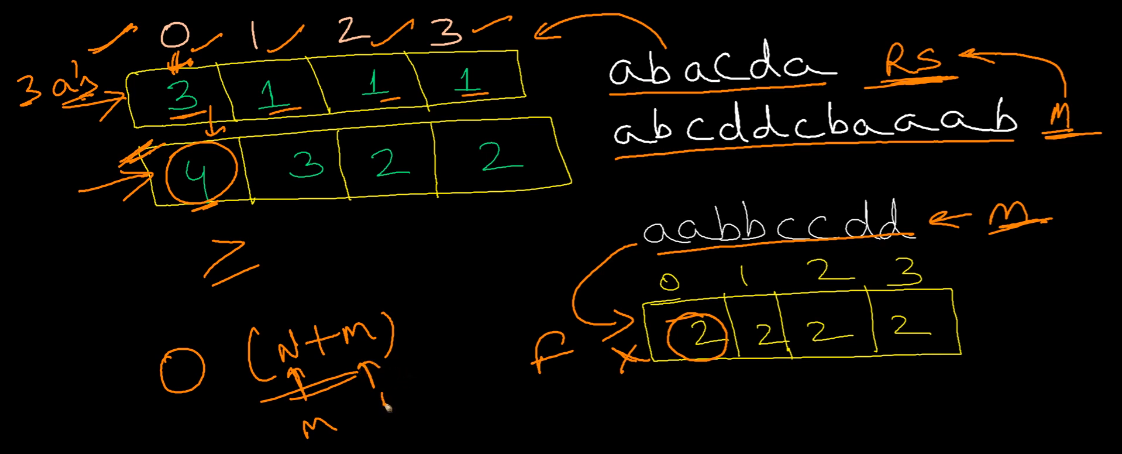
**383. Ransom Note**

Given two strings ransomNote and magazine, return true*if*ransomNote*can be constructed by using the letters from*magazine*and*false*otherwise*.

Each letter in magazine can only be used once in ransomNote.

Sol:

Will take a count of the each char of the magazine. While iterating ransomchar we will check if char is present in the magazine or not. If is present we should go to next char and delete that char so that we will not be able to use that char again.



We need to check one more scenario …if magazine char count is less that ransomeNote char count then it is not possible to create the ransome note.

**1047. Remove All Adjacent Duplicates In String**

You are given a string s consisting of lowercase English letters. A **duplicate removal** consists of choosing two **adjacent** and **equal** letters and removing them.

We repeatedly make **duplicate removals** on s until we no longer can.

**Input:** s = "abbaca"

**Output:** "ca"

**Sol:**

1) take one empty char array.

2) iterate the input char one by one and if its top char is equal to current char then we will remove the char from the result char array.

3) If it is not equal then we will add that char to result char array.

4) To delete the char, we will simply decrement index i-- of the char and for to add the char to result char array first we will add the current char and then we will increment i++.

**459. Repeated Substring Pattern**

Given a string s, check if it can be constructed by taking a substring of it and appending multiple copies of the substring together.

**Input:** s = "abab"

**Output:** true

**Explanation:** It is the substring "ab" twice.

**Sol:**

Here ab is repeated twice. So this is the logic.

1) We will find the string of 1, 2, 3……and so on…

2) check if the length of the string is divisible by length of substring.

3) If yes we will find out the number how may times.

4) Once we get the repeat count, we will create a string and append the substring that many times.

5) And then we will check if it is equal to original string or not. If it is then we will return true else false.

**821. Shortest Distance to a Character**

Given a string s and a character c that occurs in s, return *an array of integers*answer*where*answer.length == s.length*and*answer[i]*is the****distance****from index*i*to the****closest****occurrence of character*c*in*s.

The **distance** between two indices i and j is abs(i - j), where abs is the absolute value function.

**Input:** s = "loveleetcode", c = "e"

**Output:** [3,2,1,0,1,0,0,1,2,2,1,0]

**Explanation:** The character 'e' appears at indices 3, 5, 6, and 11 (0-indexed).

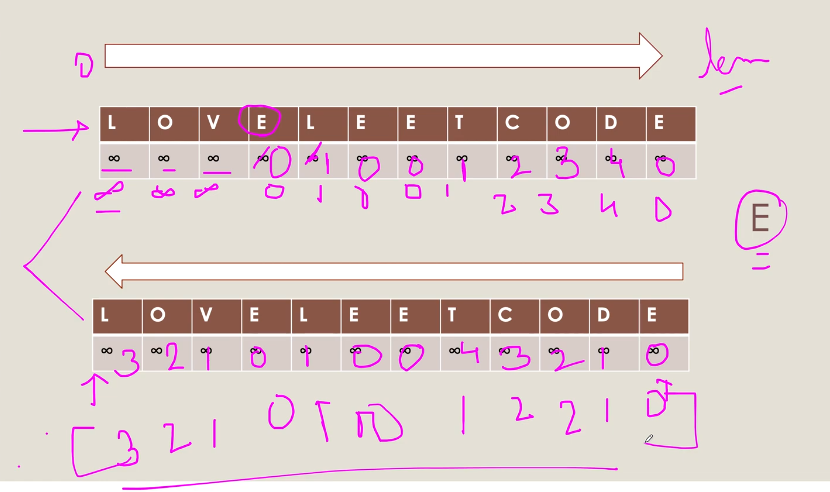
The closest occurrence of 'e' for index 0 is at index 3, so the distance is abs(0 - 3) = 3.

The closest occurrence of 'e' for index 1 is at index 3, so the distance is abs(1 - 3) = 2.

For index 4, there is a tie between the 'e' at index 3 and the 'e' at index 5, but the distance is still the same: abs(4 - 3) == abs(4 - 5) = 1.

The closest occurrence of 'e' for index 8 is at index 6, so the distance is abs(8 - 6) = 2.

**Sol:**



1) Take 2 int array, one for left and one for right and initialize it with int max value.

2) Iterate left to right the given input

a) while iterating if we find the given char then we update the value to 0

b) if char != givenChar then we simply increment the count if it is not int\_max

3) iterate the given array once more, and this time from right to left

a) perform the same operation like in the above steps

4) Minimum of both array of the each char will give the result.