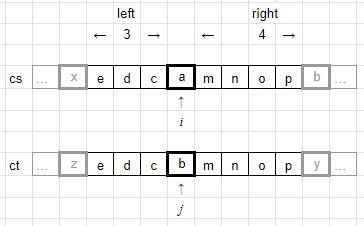
<https://leetcode.com/problems/count-substrings-that-differ-by-one-character/discuss/918183/Java-3ms-or-easy-to-understand-code>



Since the substring need to have one and only one different character, we start with finding differenct characters in String s and String t. Then, we work both backward and forward to find longest matching distance: left and right. In this example, we can choose 0 to 3 characters from left, that has 4 choice. We can also choose 0 to 4 characters from right, that has 5 choice. The total combination with only 1 different character produced by characters in position i and j is (left+1) \* (right + 1) = 4 \* 5 = 20. Sum all the combination at different i and j would give us result.

public int countSubstrings(String s, String t) {

char[] cs = s.toCharArray();

char[] ct = t.toCharArray();

int total = 0;

for (int i = 0 ; i < cs.length; ++i) {

for (int j = 0 ; j < ct.length ; ++j) {

if (cs[i] == ct[j]) continue;

int left = 0;

int x = i-1, y = j-1;

while(x >= 0 && y >= 0) {

if (cs[x--] == ct[y--])

++left;

else

break;

}

int right = 0;

x = i+1;

y = j+1;

while(x < cs.length && y < ct.length) {

if (cs[x++] == ct[y++])

++right;

else

break;

}

total += (left + 1) \* (right + 1);

}

}

return total;

}