It can be easyly stated that at first we have to count the frequencies of each character in the string. Afte change the equal frequencies so that all become unique. But doing this in the most optimesd way and u process which will be discussed here.

- So as stated first step is to find the frequencies of the characters the string. Which can be done us vector/ array of size 26. (As there are 26 characters in alpabet which can be in the string). Time Ta = O(26) which is constant i.e. O(1).
- Now we can't increase the frequency of any character in the string as only deletion is allowed. So frequencies from greatest to smallest. So sort the array. Time Taken = O(26 log 26) which is const now = O(N).

  Note : We have to only deal with the frequencies not the characters. So position of the frequencie matter as we will never need which character's frequency are dealt with.
- Now traverse the array from right to left as rightmost element is the greatest. We dont need to ch element. So start traversing fron 24th position. (As 25th is the last element). Time taken = O(25) w so total time is O(N).
- Now we will continue till frequencies become 0 for any element in the array as other elements lef break at the position where frequency is 0.
- If the current frequency (`freq[i]`) is equal or greater than the previous one (`freq[i+1`) then we current frequency less than previous one but it should be also greater than or equal to 0. (Freque 0). `So freq[i] = max(0, freq[i+1] -1)` .

  In case of doubt : So here a question may arise that why so much headache as it is sorted freq[i] freq[i+1] just do ans++ ans reduce the frequncy by 1 i.e. freq[i]--. Yes in many cases it will work b than 2 frequencies are equal then if we decrease a frequency then in next iteration the frequency previous one ans if we only reduce 1 then the answer will not correct. So we make the frequency previous and check the difference.
  Example : If frequencies be [2,2,2,2] then if we just decrease the freq by 1 en each step then next s `[2,2,1,2], del = 1 -> [2,1,1,2], del = 2 -> [1,1,1,2], del = 3` which is not the desirec difference is taken the result becomes `[2,2,2,2] -> [2,2,1,2], del = 1 -> [2,0,1,2], del` `del = 5` which is the desired ans.
- But we also need the previous value of the frequency so that we can calculate the number of freq characters deleted. So we store the previous freq as `prev` before the step mentioned before.
- So we add the reduced freq to the ans as `del += prev - freq[i]` (del is the answer).
- Return `del` at the end which will contain the total number of frequency decreased i.e. total chara

Example :

```
Let the string be  : abbccdddeeffffggg
So count of a=1, b=2, c=2, d=3, e=2, f=4, g=3. So freq = {1,2,2,3,2,4,3,0,.....,0}
So after sorting the elements of the freq array are = {0,....., 0, 1,2,2,2,3,3,4}
```