K ANAGRAMS

28 April 2022 00:24

https://pepcoding.com/resources/data-structures-and-algorithms-in-java-levelup/hashmap-and-heaps/k-anagrams-official/ojquestion
17th april 2022 - ss

Question is - Two strings are given, find that weather these two strings can be anagram, if we make maximum upto k changes in any one of the string?

Ex -
fodr --> s1
Gork --> s2
2 --> k

Output --> true

Approach - 1. Create hash map for string s1.
2. Then reduce the frequency of characters if it is available in string s2.
2. Then count the no. Of frequencies of vector v1. If it is less than or equal to k. Then it returns "true" else "false".

```
#include<bits/stdc++.h>
using namespace std;
int isdifference(vector<int>v1)
{
   int count=0;
   for(int i=0;i<26;i++)
   {

        count=count +v1[i];

   }

  // cout<<count<<endl;
   return count;
}
bool kanagram(string s1, string s2,int k)
{
  if(s1.size()!=s2.size())
  {
     cout<<"1"<<endl;
     return false;
  }
   vector<int>v1(26,0);

  for(int i=0;i<s1.size();i++)
  {
     v1[s1[i]-'a']++;
```

Function to find how many different characters are their

Function to check k anagram

If size of two strings are different, then ofcourse
It is not possible to make anagrams.

Create a hash map for string s1.

```
    for(int i=0;i<s1.size();i++)
    {
       v1[s1[i]-'a']++;


    }
      for(int i=0;i<s1.size();i++)
    {

       if(v1[s2[i]-'a']>0)
       v1[s2[i]-'a']--;

    }


    if(isdifference(v1)<=k)
    {
       return true;
    }

    return false;
}


int main()
{
    string s1,s2;
    cin>>s1>>s2;

    int k;
    cin>>k;

    bool ans = kanagram(s1,s2,k);

    if(ans)
    {

       cout<<"true"<<endl;
    }
    else
    {
       cout<<"false"<<endl;
    }
}
```

Create a hash map for string s1.

In that hashmap, for string s2, if character is
Already present, then reduce its frequency(It means that character is already balanced).
If freq. Is zero, then don't reduce it. It will impact wrongly

If isdifference is less or equal to k, it means true