Finding Subsequence

locked

by nabila_ahmed

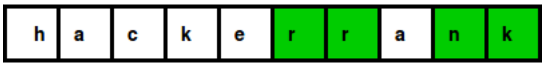
Problem Submissions Leaderboard Discussions

You have a string s and an integer k. You have to find another string t which have the following properties:

- t must be a subsequence of s.
- Every character in t must occur *at least* k times.
- t must be lexicographically largest.

Your task is to find and print the string t.

For example, let's say the string is $s =$ `hackerrank` and $k = 1$.



h	a	c	k	e	r 🟩	r 🟩	a	n 🟩	k 🟩

The solution for this is $t =$ `rrnk`. Here t is a subsequence of k and contains the characters r, n and k, each of them occurs at least $k = 1$ times.

Input Format

The first line conatins a string s denoting the original string.
The second line contains an integer k.

Constraints

- $1 \leq |s| \leq 10^5$
- $1 \leq k \leq 10^5$
- String s will only contain lowercase english letters.
- Input will be such that for every input there is a valid solution.

Output Format

Print the string t on a single line.

Sample Input 0

```
hackerrank
1
```

Sample Output 0

```
rrnk
```

Explanation 0

Here all possible subsequences are valid but `rrnk` is the lexicographically largest one.

Sample Input 1

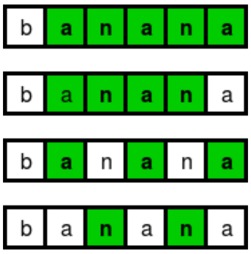
```
banana
2
```

Sample Output 1

```
nn
```

Explanation 1

Here are four possible subsequences where each chracter exists at least $k = 2$ times:



From the above subsequences, `nn` is the laxicographically largest.

Submissions: 1134
Max Score: 30
Difficulty: Medium

Rate This Challenge:
☆☆☆☆☆

More

Current Buffer (saved locally, editable)

C#

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
class Solution {

    static string solve(string s, int k) {
        // Complete this function

         Dictionary<char, int> frec = new Dictionary<char, int>();
            foreach (char ch in s)
            {
                if (frec.ContainsKey(ch))
                {
                    frec[ch]++;
                }
                else
                {
                    frec[ch] = 1;
                }
            }


            Dictionary<char, List<int>> indices = new Dictionary<char, List<int>>();
            for (int i = 0; i < s.Length; i++)
            {
                if (indices.ContainsKey(s[i]))
                {
                    indices[s[i]].Add(i);
                }
                else
                {
                    indices[s[i]] = new List<int>();
                    indices[s[i]].Add(i);
                }
            }


            char[] arr = s.ToCharArray();
            Array.Sort(arr);
            string concat = "";

            int indice = -1;

            int busca = arr.Length - 1;
            while (frec[arr[busca]] < k)
            {
                busca--;
            }
            indice = indices[arr[arr.Length - 1]][0];
            concat += arr[arr.Length - 1];

            //Console.WriteLine(indice);

            for (int i = arr.Length - 1; i >= 0; i--)
            {

                    List<int> listaIndices = indices[arr[i]];
                    for (int j = 0; j < listaIndices.Count; j++)
                    {
                        if (listaIndices[j] > indice)
                        {
                            concat += arr[i];
                            indice = listaIndices[j];
                        }
                    }

            }

            Dictionary<char, int> frecconcat = new Dictionary<char, int>();
            foreach (char ch in concat)
            {
                if (frecconcat.ContainsKey(ch))
                {
                    frecconcat[ch]++;
                }
                else
                {
                    frecconcat[ch] = 1;
                }
            }

            string ans = "";
            foreach (char ch in concat)
            {
                if (frecconcat[ch] >= k)
                {
                    ans += ch;
                }
            }


            return ans;


    }

    static void Main(String[] args) {
        string s = Console.ReadLine();
        int k = Convert.ToInt32(Console.ReadLine());
        string result = solve(s, k);
        Console.WriteLine(result);
    }
}

```

Line: 93 Col: 24

Upload Code as File

□ Test against custom input

Run Code

Submit Code

Join us on IRC at #hackerrank on freenode for hugs or bugs.

Contest Calendar | Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature