CODE FESTIVAL 2016 予選 B 解説

DEGwer

2016 年 10 月 10 日

A: Signboard

問題概要

16 文字の文字列が与えられる。最小で何文字変えれば”CODEFESTIVAL2016”になるか。

解説

与えられた文字列と、文字列”CODEFESTIVAL2016”を持っておき、同じ箇所の文字を比較して異なるところの個数を単純なループで数えればよいです。

B: Qualification Simulator

問題概要

N 人の参加する予選が行われ、最大で $A+B$ 人が通過する。海外の学生はその中で B 位まで、国内の学生は海外の学生と合わせて $A+B$ 位までに入れば予選を通過できる。学生でない人は通過できない。各人について、予選を通過したかどうかを出力せよ。

解説

カウンタを 2 つもち、片方ではこれまでに通過した人数、もう片方ではこれまでに通過した海外の学生の人数を持っておきます。

上位から順に、

- 学生でないなら通過せず、
- 国内の学生なら、前者のカウンタ値が $A+B$ 以下なら通過し、そのカウンタを 1 増加させ、そうでなければ通過せず、
- 海外の学生なら、前者のカウンタ値が $A+B$ 以下かつ後者のカウンタ値が B 以下なら通過し、両方のカウンタを 1 増加させ、そうでなければ通過しない

として、順次通過するかどうかを求めていけばよいです。

C: Gr-idian MST

問題概要

$(W+1) \times (H+1)$ のグリッドグラフがある。(i,j) と $(i+1,j)$ を結ぶときは j によらずコストが p_i、(i,j) と $(i,j+1)$ を結ぶときは i によらずコストが q_j かかる。最小全域木のコストを求めよ。

$W, H \leq 10^5$

解説

最小全域木をクラスカル法で求めることを考えます。クラスカル法では、辺をコストの小さい順にソートし、両端点が異なる連結成分にあるならその辺を使うという操作を繰り返します。

それを行うために、p_i たちと q_j たちを一緒にしてソートします。これらの値の小さい順に、辺を追加できるだけ追加していけば最小全域木が求まります。

では、各ステップで辺は何本追加できるでしょうか。配列の p_i に対応する要素を見るときには、(i,j) と $(i+1,j)$ を結ぶ辺を何本追加できるかを求め、さらにすべての j に対し、(i,j) と $(i+1,j)$ を連結にすればよいです。

そしてこの連結操作は、「これ以降 x 座標 i と $i+1$ を同一視する」という操作です。よって、この操作では「x 座標 j を固定したときに、(i,j) と $(i,j+1)$ を結ぶような辺の数が 1 減少する」ということが起こります。

以上をまとめると、

- p_i たちと q_j たちをまとめてソートし、
- $a = W+1, b = H+1$ として、その列を前から見て、
- (i,j) と $(i+1,j)$ を結ぶ辺を見るときはそのような辺を b 本使い、a を 1 減らし、
- (i,j) と $(i,j+1)$ を結ぶ辺を見るときはそのような辺を a 本使い、b を 1 減らす

という操作で満点を得ることができます。計算量は $O((W+H)\log(W+H))$ です。

D: Greedy Customers

問題概要

N 項からなる数列がある。正整数 k を指定し、k 以上の最初の要素から k を引くという操作を、数列のどの要素も 0 になってはいけないという条件下で最大で何回できるか。$N \leq 10^5$

解説

$k = 1$ は、最大で (最初の要素 -1) 回指定することができます。それ以降 1 は選択できません。もし数列に 2 が存在しないのならば、それ以降は数列の 2 以上の最初の要素が奇数なら 2 を、偶数なら 3 を選択し続けることで、明らかに最大の回数を達成します。(1 は選択できないため)

では、数列に 2 が存在する場合はどうなるでしょうか。数列の 2 以上の最初の要素が 2 のとき、それ以降 1 も 2 も選択することはできないため、それ以降は 3 以上の整数を指定する必要があります。逆に、それまでは 2 を選択できることになります。

より一般に、最初の要素を 1 まで減らした後、ある位置 x までの位置に整数 $2, 3, ..., t$ がこの順に (必ずしも連続せず) 出現しているとします。このとき、位置 x 以降の要素を減らすためには、そのあとどんな操作を行っても、t 以下の整数を指定することはできません。これは、t 以下の整数 s を指定するためには x までに現れる s を減らさなければなりませんが、そのためには s 以下の整数を指定せねばならず... となり、数列の要素が 0 になることを避けられないからです。

以上より、次のように整数を減らしていくのが最適です。

- まず、最初の要素を 1 になるまで減らす
- 次に 2 が現れるまで、すべての要素を 2 ずつ (ただし偶数は 3) 減らす
- 次に 3 が現れるまで、すべての要素を 3 ずつ (ただし 3 の倍数は 4) 減らす
- $4, 5, 6, ...$ と同じように繰り返す

このアルゴリズムは容易に $O(N)$ 時間でシミュレートすることができるので、時間計算量 $O(N)$ でこの問題を解くことができます。

E: Lexicographical disorder

問題概要

英小文字からなる文字列が N 個与えられる。文字どうしの大小関係を表す順列と、整数 k の組が与えられるので、文字列たちをその順序で辞書順にソートしたとき k 番目の文字列が何番目に来るかを求めるクエリを Q 個処理せよ。

$N, Q \leq 10^5$, 文字列長の合計 $\leq 4 \times 10^5$

解説

文字列たちをある文字同士の大小関係のもとでソートしたときに辞書順で x 番目であるというのは、辞書順でその文字列より小さい文字列が $x-1$ 個あるということと等価なので、その文字列より辞書順で小さい文字列の個数を数えることにします。

文字列 s が文字列 t より辞書順で小さいというのは、以下のいずれかの状況です。

- s は t の接頭辞である
- s と t を前から見て、初めて異なる文字が現れるようなところの文字を比較したときに、s の文字のほうが小さい

前者の個数は文字同士の大小関係によらないので、全ての文字列についてあらかじめ trie 木上での簡単な操作で求めておくことができます。これ以降、後者を考えます。

「初めて異なる文字が現れる位置」は文字同士の大小関係によりません。よって、t より小さい t の接頭辞でない文字列がいくつあるかは、これらの「初めて異なる文字が現れる位置の文字」を、ほかのすべての文字列と比較すればいいことになります。

さて、この比較はどれくらい行えばよいでしょうか。文字は 26 種類しかないので、比較すべき文字の組というのは高々 26^2 通りしかありません。よって、各文字列 t と、文字の組 (x_1, x_2) に対し、「$x_1 < x_2$ のとき、その条件で t より小さいことが決まるような文字列の個数」というのをあらかじめ計算しておけば、クエリ当たり 26^2 回の比較を行い、それによって得られた値をすべて足し合わせることで、すべてのクエリにこたえることができます。

この前計算は、trie 木を使って ($\sigma = 26$ として)、O(文字列長の合計 $\times \sigma$) で行うことができるので、合計 O(文字列長の合計 $\times \sigma + Q\sigma^2$) でこの問題を解くことができました。

CODE FESTIVAL 2016 Qual B Editorial (English)

October 10, 2016

A : Signboard

Problem Summary

We have a 16-letter string. How many letters do we need to change in order to turn it into "CODEFESTIVAL2016"?

Solution

Store two strings, the given one and "CODEFESTIVAL2016". Then, count the number of the positions where the two strings have different letters, in a single loop.

B : Qualification Simulator

Problem Summary

N participants took part in the qualification contest. At most $A + B$ students can qualify. A Japanese student can qualify if he/she ranks $(A + B)$-th or above among all students. In addition to this requirement, an oversea student must also rank B-th or above among the overseas students to qualify. A non-student participant cannot qualify. For each contestant, determine whether he/she qualifies.

Solution

Maintain two counters, one for the number of all students who have already qualified (let this counter be counter 1), the other for the number of oversea students who have already qualified (let this counter be counter 2).

Examine the participants from the top rank down, and for each participant, determine whether he/she qualifies as follows:

- if the current participant is non-student, he/she does not qualify.
- if the current participant is a Japanese student, he/she qualifies if and only if the value of counter 1 is smaller than $A + B$. If he/she qualifies, we will increment the value of counter 1 by 1.
- if the current participant is an oversea student, he/she qualifies if and only if the value of counter 1 is smaller than $A + B$ and the value of counter 2 is smaller than B. If he/she qualifies, we will increment the values of both counters by 1.

C : Gr-idian MST

Problem Summary

We have a $(W+1) \times (H+1)$ grid graph. The cost of the edge connecting (i, j) and $(i+1, j)$ is p_i regardless of j, and the cost of the edge connecting (i, j) and $(i, j+1)$ is q_j regardless of i. Find the cost of a minimum spanning tree of this graph.

$W, H \leq 10^5$

Solution

We will find the minimum spanning tree by Kruskal's algorithm. In this algorithm, the edges are sorted in order of increasing cost, then the following is repeated: use an edge if it connects two vertices from different connected components.

For this problem, we unite all p_i and q_j into an array and sort it in order of increasing cost. Then, for each of these elements, we will try to use as much corresponding edges as possible to obtain a minimum spanning tree.

How many edges can we use in each of these steps? When we are examining an element that corresponds to p_i, we will find the number of available edges connecting (i, j) and $(i+1, j)$, then for all j, connect (i, j) and $(i+1, j)$.

This action of connecting (i, j) and $(i+1, j)$ can be considered as follows: "from now on, two x coordinates i and $i+1$ are merged and regarded as the same coordinate." Thus, after this action, the number of edges connecting (i, j) and $(i, j+1)$, when the y coordinate j is fixed, decreases by one.

Therefore, the answer can be computed as follows:

- Unite all p_i and q_j into an array and sort it in order of increasing cost.
- Let $a = W+1, b = H+1$. Examine each element of the array in order.
- When we are looking at edges connecting (i, j) and $(i+1, j)$, use b of such edges, then decrement a by one.
- When we are looking at edges connecting (i, j) and $(i, j+1)$, use a of such edges, then decrement b by one.

The time complexity of this solution is $\mathrm{O}((W+H)\log(W+H))$.

D : Greedy Customers

Problem Summary

We have a sequence of N integers. Consider the following operation: specify an positive integer k, then subtract k from the first element in the sequence that is k or above. Under the condition that no element should become zero, find the maximum number of times the operation can be performed.

$N \leq 10^5$

Solution

$k = 1$ can be specified at most ((the first element) $- 1$) times. After $k = 1$ is chosen this number of times, $k = 1$ is no longer available. If the integer 2 does not exist in the sequence, we can achieve the maximum number of operations by repeating the following: choose $k = 2$ if the first element that is 2 or above, is odd, and choose $k = 3$ otherwise.

What about the case where the integer 2 does exist in the sequence? When the first element that is 2 or above, is 2, we can choose neither $k = 1$ nor $k = 2$, thus we are forced to specify $k = 3$ or above. (In another perspective, $k = 2$ is available until then.)

More generally, suppose that the integers $2, 3, \ldots, t$ occurs up to the position x in the sequence (not necessarily consecutive), when the first element in the sequence is already decremented to 1. In order to affect and decrement an element that occurs after the position x, we cannot specify an integer that is t or below, no matter what operations are performed afterwards. This is because, in order to specify an integer $s \leq t$, we need to affect and decrement the integers s that occur up to the position x, which requires us to specify an integer that is s or below, which is impossible without breaking the condition that no element should become zero.

Therefore, the optimal strategy is as follows:

- Specify $k = 1$ until the first element in the sequence becomes 1.
- Then, specify $k = 2$ (or $k = 3$ if the element being affected is odd) until the element being affected becomes 2.
- Then, specify $k = 3$ (or $k = 4$ if the element being affected is a multiple of 3) until the element being affected becomes 3.
- Continue in this manner through the sequence.

This strategy can be simulated in $\mathrm{O}(N)$ time.

E : Lexicographical disorder

Problem Summary

We have N strings $S_1, S_2, \ldots, S_N$ of lowercase alphabetical letters. Process Q queries of the following form:

Query: you are given a permutation that specifies the magnitude relation of the alphabetical letters, and an integer k. Suppose that when the N strings are sorted according to the specified magnitude relation of the letters, the string S_k comes x-th in the resulting sequence. Find x.

$N, Q \leq 10^5$, (the total length of S_i) $\leq 4 \times 10^5$

Solution

What we must find in each query, is the number of the strings that is smaller than S_k according to the specified magnitude relation, plus one.

A string s is lexicographically smaller than another string t, if and only if one of the following condition is satisfied:

- s is a prefix of t.
- s has a smaller letter than t, at the leftmost position where s and t have different letters.

Consider the first condition. For any of the strings, the number of the strings that are prefixes of it does not depend on the magnitude relation of the letters, and can be computed beforehand using trie.

Regarding the second condition, "the leftmost position where s and t have different letters" does not depend on the magnitude relation of the letters. Thus, in each query, we can compare the letters at those positions with the other strings, to find the number of the strings that are smaller than the specified string but not prefixes of it.

Since the alphabet has 26 letters, we have at most 26^2 pairs of letters to compare. For each string t and each pair of letters (x_1, x_2), we compute the following beforehand: the number of the string that will be smaller than t when the condition $x_1 < x_2$ holds. Then, we can process each query by performing 26^2 comparisons and find the total number of the strings that is smaller than the specified string.

This computation beforehand can be performed in O((the total length of S_i) $\times \sigma$) time using trie (where $\sigma = 26$), therefore the whole problem can be solved in O((the total length of S_i) $\times \sigma + Q\sigma^2$) time.