AtCoder Beginner Contest 094 / AtCoder Regular Contest 095 解説

writer: semiexp, nuip

2018 年 4 月 14 日

For International Readers: English editorial starts on page ?.

A: Cats and Dogs

猫の数は，B 匹すべてが犬のときに最小 (A 匹) になります．一方，B 匹すべてが猫のときに最大 ($A+B$ 匹) になります．

よって，X が A 以上 $A+B$ 以下かどうか判定すればよいです．

- C++ による解答例: `https://abc094.contest.atcoder.jp/submissions/2343785`
- Python による解答例: `https://abc094.contest.atcoder.jp/submissions/2344330`

B: Toll gates

ゴールするまでに，元いたマスへ引き返すことをする必要はありません．すなわち，マス X からマス $X-1, X-2, \ldots, 0$ と進むか，$X+1, X+2, \ldots, N$ と進むかのいずれかです．この 2 パターンそれぞれについて，通る料金所の個数を数え，最小値を求めればよいです．

- C++ による解答例: `https://abc094.contest.atcoder.jp/submissions/2343740`
- Python による解答例: `https://abc094.contest.atcoder.jp/submissions/2344333`

C: Many Medians

入力をソートしたものを $Y_1 \leq Y_2 \leq \cdots \leq Y_N$ とします．このとき B_i を考えます．ソートした後に A_i が l_i 番目に現れるとします（すなわち $Y_{l_i} = A_i$ が成り立ちます）．B_i は $Y_1, \ldots, Y_{l_i-1}, Y_{l_i+1}, \ldots, Y_N$ の中央値であることに注意します．すると，$Y_1 \leq \cdots \leq Y_{l_i-1} \leq Y_{l_i+1} \leq \cdots \leq Y_N$ より，この $N-1$ 個の値のうち $\frac{N}{2}$ 番目の値を求めればよいことがわかります．これは，$l_i \leq \frac{N}{2}$ であれば $Y_{N/2+1}$，$l_i \geq \frac{N}{2}+1$ であれば $Y_{N/2}$ になることが確かめられます．

D : Binomial Coefficients

$\mathrm{comb}(n, r)$ を並べた表はパスカルの三角形 (https://ja.wikipedia.org/wiki/と呼ばれています。これを観察すると下に行くほど（n が大きくなると）値が大きくなることが分かります。また、同じ段なら真ん中よりのほうが値が大きくなることが分かります。このことから、$a_1, a_2, \ldots, a_n$ のうち最大のものを n として使い、n 以外で $n/2$ に最も近いものを r として使うと最大になることが予想できます。

この予想が正しいことを示します。次の事実を使います。

$$\mathrm{comb}(n, r) = \frac{n(n-1)(n-2)\ldots(n-r+1)}{r(r-1)(r-2)\ldots 1}$$

まず、最初の予想が正しいことを確かめるために次を示します。

主張. $r > 0$ のとき、$\mathrm{comb}(n+1, r) > \mathrm{comb}(n, r)$

証明. $r = 1$ のときは明らか。$r > 1$ とする。

$$\begin{aligned}
&\mathrm{comb}(n+1, r) - \mathrm{comb}(n, r) \\
=&\frac{(n+1)n(n-1)\ldots(n-r+2)}{r(r-1)(r-2)\ldots 1} - \frac{n(n-1)(n-2)\ldots(n-r+1)}{r(r-1)(r-2)\ldots 1} \\
=&\frac{n(n-1)\ldots(n-r+2)}{r(r-1)(r-2)\ldots 1}\left((n+1) - (n-r+1)\right) \\
=&\frac{n(n-1)\ldots(n-r+2)}{r(r-1)(r-2)\ldots 1} r > 0
\end{aligned}$$

□

次に、二番目の予想が正しいことを示すのには、n 個から r 個選ぶのと n 個から選ばない $n-r$ 個を決めるのが同じであることを踏まえると、次を確かめれば十分です。

$$r + 1 \leq n/2 \text{ のとき、} \mathrm{comb}(n, r+1) > \mathrm{comb}(n, r)$$

これは $\frac{\mathrm{comb}(n,r+1)}{\mathrm{comb}(n,r)}$ を計算すれば分かります。

E: Symmetric Grid

行の入れ替えの操作と，列の入れ替えの操作は，交換可能であることに注意します．よって，最初に何回か行の入れ替えの操作のみを行った後，何回か列の入れ替えの操作のみを行うとしてもかまいません．

まず，行の入れ替えの方法を固定して，その後列の入れ替え操作のみでマス目を点対称的にできるかどうか判定することを考えます．$j = 1, 2, \ldots, W$ に対して，点対称的なマス目の j 列目は，$W + 1 - j$ 列目を上下反転したものになっていることに注意します．特に，W が奇数のとき，$(W + 1)/2$ 列目は上下対称になります．逆に，この条件が成り立てば，マス目は点対称的になります．

よって，これは次の方法で判定することができます．(実際の実装では，具体的に点対称的にする方法を構成する必要はありません）

- はじめ，すべての列を未使用としておく．盤面も，すべての列を取り出して，空にしておく．
- 未使用の列がある場合，それを上下反転したような未使用の他の列が存在するか判定する．
  - もし存在すれば，その 2 つの列を使用済みにし，まだ埋まっていない最も左，最も右の列にそれぞれ配置する．
  - 存在しない場合，もし W が偶数ならば，点対称的にすることは不可能である．W が奇数ならば，その列が上下対称か判定し，対称ならばその列を $(W + 1)/2$ 列目に置く．すでに $(W + 1)/2$ 列目が埋まっているか，対称でないならば，点対称的にすることは不可能である．

このアルゴリズムは $O(HW^2)$ で動作します.（列をソートするなどの方法で，$O(HW \log W)$ にすることもできます）

次に，行の入れ替えの方法を考えます．単純に考えると，行の入れ替えの方法は $H!$ 通りあり，大きすぎます．ここで，$1, H$ 行目，$2, H-1$ 行目，... のように行同士をペアにして考えると (H が奇数の場合，ペアにならない行が 1 行だけ存在します)，ペア関係が同一な入れ替え方法では，点対称的にできるかどうかは全く同じになることがわかります．なので，行同士をペアにする異なる方法をすべて試し，それぞれの方法に対しては適当に行の入れ替えをしてから，先の列の入れ替えによる判定を行うことで，この問題を解くことができます．ペアにする方法は，最大で $11!! = 11 \times 9 \times 7 \times 5 \times 3 \times 1 = 10395$ 通りしかないので，十分この問題を解くことができます．

F : Permutation Tree

高橋くんの能力は言い換えると次のようになります。
max を -1 として、p_i の小さい順に次の処理をする。

- $p_i \neq 1$ のとき、max と i を辺でつなぐ
- $i >$ max であれば、 max を i で書き換える

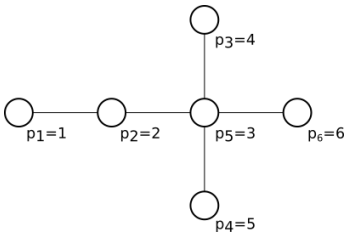
つまり、今までに出てきた添字のうちの最大のところに辺を貼っていく操作になります。添字の最大が更新された頂点だけに注目すると、これらの頂点は p_i の小さい順に頂点を並べたパスをなしていることが分かります。一方で、最大が更新されていない頂点は、このパスに含まれる頂点と直接つながることが分かります。このようなグラフは caterpillar(`https://en.wikipedia.org/wiki/Caterpillar_tree`) と呼ばれています。先程言及したパスのことを今後胴体と呼び、胴体に含まれていない頂点のことを脚と呼ぶことにします。
以上で、高橋くんの能力で作れるグラフは必ず caterpillar であることが分かりました。実は caterpillar であれば必ずその木を作れるような順列が存在します。

caterpillar 判定

与えられた木が caterpillar であるかどうかを判定する方法は次のとおりです。まず、木の直径を求めます (`http://www.prefield.com/algorithm/graph/tree_diameter.html`)。与えられた木が caterpillar である場合、この木の直径を胴体とみなすことができます。これは脚の長さが 1 であることから明らかです。胴体が求まったので、あとは残りの頂点が直接胴体に繋がれているか判定するだけです。

順列の生成

次に、与えられた caterpillar を作る順列の作り方を考えます。胴体が左右に並んでいて、左から右に添字も値も増加しているとします。胴体の頂点の添字を左から $b_1, \ldots, b_l$ とします。



胴体のうちのある頂点 b_i に注目します。この頂点から脚が k_i 本生えているとき、その k_i 個の頂点は頂点 b_i よりも値が大きくて、添字が小さい必要があります。この条件の元で辞書順で小さくなるよう気をつけると、次のようにしてできる順列が答えの候補になります。

$(1, 2, \ldots, n)$ という順列から始める。
各 $i = 1, 2, \ldots, l$ について、左から $\sum_{j=0}^{i-1}(k_i + 1)$ 番目の数を k_i こ分右にずらす。

caterpillar のとり方と答え

与えられた木を caterpillar とみなす時、胴体にあたるパスの取り方は複数通りありえます。上記の caterpillar の構築方法をふまえると、胴体はできるだけ長くなるように取った方がいいことが分かります。胴体の向きが 2 通りありますが、これは両方試して辞書順で小さい順列が得られたほうを答えとすれば良いです。

AtCoder Beginner Contest 094 / AtCoder Regular Contest 095 Editorial

writer: semiexp, nuip

April 14th, 2018

A: Cats and Dogs

- C++ example: `https://abc094.contest.atcoder.jp/submissions/2343785`
- Python example: `https://abc094.contest.atcoder.jp/submissions/2344330`

B: Toll gates

- C++ example: `https://abc094.contest.atcoder.jp/submissions/2343740`
- Python example: `https://abc094.contest.atcoder.jp/submissions/2344333`

C: Many Medians

Let $Y_1 \leq Y_2 \leq \cdots \leq Y_N$ be the sorted list of X. Suppose that $Y_{l_i} = A_i$. B_i is the median of $Y_1, \ldots, Y_{l_i-1}, Y_{l_i+1}, \ldots, Y_N$. Thus, B_i is the $\frac{N}{2}$-th value of $Y_1, \ldots, Y_{l_i-1}, Y_{l_i+1}, \ldots, Y_N$. This is $Y_{N/2+1}$ if $l_i \leq \frac{N}{2}$, and $Y_{N/2}$ if $l_i \geq \frac{N}{2} + 1$.

D : Binomial Coefficients

We shuold choose the maximum of $a_1, a_2, \ldots, a_n$ as n because for a fixed r, the function $comb(n, r)$ is monotonously increasing. This follows from the following fact:

$$\text{If } r > 0,\ \mathrm{comb}(n+1, r) > \mathrm{comb}(n, r)$$

(because $\mathrm{comb}(n+1, r) = \mathrm{comb}(n, r) + \mathrm{comb}(n, r-1)$)
Once we choose n, we should choose r that is closest to $n/2$. This follows from the following fact:

$$\text{If } r + 1 \leq n/2,\ \mathrm{comb}(n, r+1) > \mathrm{comb}(n, r)$$

Compute $\frac{\mathrm{comb}(n,r+1)}{\mathrm{comb}(n,r)}$ to prove this.

E: Symmetric Grid

Notice that an operation about rows and an operation about columns are commutative. Thus, assume that we first perform operations about rows, and then operations about columns.

Suppose that we finished operations about rows. How can we check if we can achieve a goal by operations about columns? Notice that, for each $j = 1, 2, \ldots, W$, the j-th column must be the reverse of the $W + 1 - j$-th column. In particular, when W is odd, $(W + 1)/2$ is a palindrome.

Therefore, we can use the following method do check if we can achieve a goal by operations about columns:

- Initially, all rows are "unused".
- Choose a particular unused row. Reverse it, and check if it matches with another unused row.
  - If it matches with another row, mark the two rows as "used".
  - Otherwise, if W is even, the answer is "NO". If W is odd and the current row is not a palindrome, the answer is "NO". If W is odd and the current row is a palindrome, we should put it in the $(W + 1)/2$-th row. If this row is already filled, the answer is "NO".

It works in $O(HW^2)$ time (or by binary search you can do it in $O(HW \log W)$)

If we consider all $H!$ possible orders of rows, it will be too slow. Instead, notice that only the set of pairs ($(1, H)$-th rows, $(2, H - 1)$-th rows, ...) matters. (In case H is odd, one row is left unpaired.)

Thus, we only need to try all pairings of rows. Since there are at most $11!! = 11 \times 9 \times 7 \times 5 \times 3 \times 1 = 10395$ ways of pairings, it's fast enough.

F : Permutation Tree

We can restate the generation of a tree as follows:

Let max $= -1$. Initially, all vertices are painted black. Perform the following operation in the increasing order of p_i (that is, first we choose an i such that $p_i = 1$ and perform the operation, and so on):

- If $p_i \neq 1$, add an edge between vertex max and vertex i.
- If $i >$ max, replace max with i, and color vertex i red.

It's clear that when you add an edge, it is always between vertex i (the current vertex) and the most recently painted red vertex. Also, when we paint a vertex, it is always the current vertex.

Thus,

- The red vertices will form a path.
- A black vertex is a leaf, and it's incident to a red vertex.

Such graphs are called caterpillar(`https://en.wikipedia.org/wiki/Caterpillar_tree`). From now on, we call red vertices "body", and black vertices "feet". Also, we call one end of the body "head", and the other end "tail". The head corresponds to the vertex with $p_i = 1$.

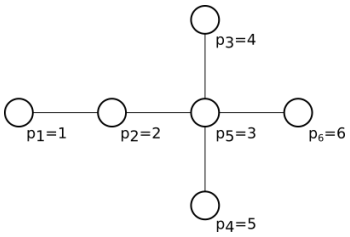
Therefore, if the given tree is not a caterpillar, we should print "-1". Otherwise, it turns out that the solution always exists, as we describe later.

Check if the given tree is a caterpillar

Find the diameter of the tree. The tree is a caterpillar if and only if it forms the body (i.e., all vertices are either in the diameter or adjacent to one vertex in the diameter).

Find the lexicographically smallest permutation that generates a given caterpillar

Let's fix a caterpillar. How can we find a permutation that generates this caterpillar? Let $b_1, \ldots, b_l$ be the vertices in the body in the order from head to tail. Let k_i be the number of feet that grow from b_i.



For example, in the picture above, $\{k_i\} = \{0, 0, 2, 0\}$.

For each foot, the index of its parent must be greater than its index. The value of its parent must be less than its value. The lexicographically smallest permutation that satisfies these conditions is the following permutation:

> Consider a permutation $(1, 2, \ldots, n)$. For each $i = 1, 2, \ldots, l$ in this order, do the following: move the $\sum_{j=0}^{i-1}(k_i + 1)$-th (from left, 1-indexed) element to the right by k_i positions.

For example, we get $1, 2, 4, 5, 3, 6$ for the example above.

It's easy to see that this permutation generates a desired caterpillar.

How to choose a head and a tail of a caterpillar

There are multiple ways to choose a head and a tail of a caterpillar. From the construction above, it's clear that we shuold maximize the length of the body, so we should choose a diameter and the head and the tail should be two ends of the diameter. There are two ways to orient the body: we should try both ways.