AtCoder Beginner Contest 101 / AtCoder Regular Contest 099 解説

writer: semiexp

2018 年 6 月 23 日

For International Readers: English editorial starts on page 5.

A: Eating Symbols Easy

高橋君が思い浮かべている整数を，S の各文字ごとに更新していけばよいです．あるいは，S の中に現れる + の個数から，- の個数を引くことでも求められます．

- C++ による解答例: https://abc101.contest.atcoder.jp/submissions/2711190
- Ruby による解答例: https://abc101.contest.atcoder.jp/submissions/2711191
- Python による解答例: https://abc101.contest.atcoder.jp/submissions/2711198

B: Digit Sums Easy

$S(n)$ を計算できればよいです．これは,「現在の n の 1 の位を求め合計値に足し込み，1 の位を取り去る」ことを繰り返せばよいです．1 の位は 10 で割った余りとして求められます．また，1 の位を取り去った値は 10 で割った商として求められます．

- C++ による解答例: https://abc101.contest.atcoder.jp/submissions/2711205
- Ruby による解答例: https://abc101.contest.atcoder.jp/submissions/2711210
- Python による解答例: https://abc101.contest.atcoder.jp/submissions/2711212

C: Minimization

数列の最小値は 1 です．どのように操作を行っても，最小値が 1 以外に変化することはありません．なので，数列の要素をすべて 1 にすることを考えます．

数列中の 1 の個数は，はじめ 1 です．この個数は，1 回の操作で $K-1$ までしか増やすことができません．なぜならば，1 の個数を増やすためには，操作で選ぶ要素の中に 1 が含まれている必要があるためです．よって，少なくとも $\lceil \frac{N-1}{K-1} \rceil$ 回の操作が必要であることがわかります．

一方，うまく操作を行うことで，$\lceil \frac{N-1}{K-1} \rceil$ 回の操作ですべての要素を 1 にすることが可能です．はじめ $A_p = 1$ であるとします．ここで，$1 + g(K-1) \leq p \leq 1 + (g+1)(K-1)$ なる整数 g をとります．このとき，次のように操作を行えばよいです：

- $i = g, g-1, g-2, \ldots, 1$ に対して，この順で要素 $i(K-1)+1, i(K-1)+2, \ldots, i(K-1)+K$ を選んで操作を行う．

- $i = g+1, g+2, \ldots, \lceil \frac{N-1}{K-1} - 1 \rceil$ に対して，この順で要素 $i(K-1)+1, i(K-1)+2, \ldots, i(K-1)+K$ を選んで操作を行う．
- 要素 $N-K+1, N-K+2, \ldots, N$ を選んで操作を行う．

D: Snuke Numbers

「N 以上の整数 n であって，$\frac{n}{S(n)}$ を最小にするもの (複数ある場合は，そのうち n が最小になるもの)」を $f(N)$ で表します．すると，$N = 1$ から始めて，$N \leftarrow f(N+1)$ という更新を順次繰り返すことで，すぬけ数を小さいほうから順に列挙することができます．以降，この $f(N)$ について考えます．

まず，$\frac{x}{S(x)}$ の $x \geq N$ での最小値が常に存在することを示しておきます．$10^{d-1} \leq n < 10^d$ のとき，n の最小値は 10^{d-1}，$S(n)$ の最大値は $9d$ であることから，$\frac{x}{S(x)} \geq \frac{10^{d-1}}{9d}$ です．この下界は $d \geq 1$ では d を 1 増加させると少なくとも 5 倍になるので，十分 d を大きくすると $\frac{N}{S(N)}$ より大きくなります．よって，このような d に対して 10^d 未満の整数 x のみを考えればよいことから，最小値は存在します．

$x \geq N$ で $\frac{x}{S(x)}$ を最小にする x (複数ある場合は，そのうち最小のもの) をとります．$x > N$ と仮定して，x と N を上位の桁から比較していったとき，初めて異なる値になる桁を 10^d の位とします．このとき，$x = 10^{d+1}\lfloor \frac{N}{10^{d+1}} + 1 \rfloor - 1$ であること（つまり，x の $10^0, \ldots, 10^d$ の位の数字がすべて 9 であること）を示します．

まず，x の $10^0, 10^1, \ldots, 10^{d-1}$ の位の数字はすべて 9 です．さもなくば，$x' = 10^{d+1}\lfloor \frac{N}{10^{d+1}} \rfloor - 1$ を考えると，$N \leq x' < x$ かつ $S(x') \geq S(x)$ であることから，x が $\frac{x}{S(x)}$ を最小にすることに反します．

よって，$x_0 = 10^d \lfloor \frac{N}{10^d} + 1 \rfloor - 1$ とおくと，ある $1 \leq w \leq 9$ が存在して，$x = x_0 + w \cdot 10^d$ かつ $S(x) = S(x_0) + w$ となります．x の最適性より，$\frac{x_0}{S(x_0)} > \frac{x}{S(x)} = \frac{x_0 + w \cdot 10^d}{S(x_0)+w}$ が成立します．よって，$x_0 > S(x_0 \cdot 10^d$ が成り立ちます．一方，同様にして，w を大きくすればするほど $\frac{x}{S(x)}$ が小さくなることもわかります．実際には x の 10^d の位が 9 を超えることはできないので，その範囲で w を大きくするのが最適です．よって，x の 10^d の位も 9 であることが示されました．

よって，x の候補として考えられるものが N 並びに $d = 0, 1, 2, ...,$ に対して $x = 10^{d+1}\lfloor \frac{N}{10^{d+1}} + 1 \rfloor - 1$ の形で書けるものに限られることがわかりました．ここで，d としては $\log_{10} N + 1$ 以下で考えれば十分です．よって，候補を十分少なく絞ることができたので，これらすべてに対して $\frac{x}{S(x)}$ を計算して比較すれば答えが得られます．

E: Independence

問題を言い換えると，次のようになります：

> グラフから，できるだけ多くの辺を取り除いて，グラフ全体がちょうど 2 つのクリークの合併になるようにする．

これを，グラフを補グラフ（グラフの辺の有無を反転して得られるグラフ）にして考えると，次のように言い換えることができます：

> グラフに，できるだけ多くの辺を加えて，グラフ全体が完全二部グラフになるようにする．

以降，補グラフについて考えることにします．

元のグラフが二部グラフでなければ，如何に辺を加えてもグラフ全体を完全二部グラフにすることは不可能です．一方，元のグラフが二部グラフの場合は，頂点集合を 2 つの disjoint な集合 S, T に分割して，S, T にまたがる辺のみが存在するようにできるので，S, T にまたがる辺すべてを追加することで，グラフ全体を完全二部グラフにすることができます．

グラフを連結成分分解すると，各成分においては頂点集合の（いずれの辺も異なる集合間にまたがるような）二分割は，集合の入れ替えを除いて一意に定まります．i 番目の連結成分の頂点集合をこのように二分割したものを S_i, T_i とし

ます．すると，S は各 i に対して S_i, T_i のちょうど片方を選んで，それらの和集合をとることで構成することができます．(T は頂点全体から S に含まれる頂点を除くことで得られます)

さて，この問題中で問われている「両端の都市が同じ州内に属しているような道の本数」は，S の大きさのみによって定まることがわかります．よって，S の大きさとしてありうるものをすべて列挙すればこの問題を解くことができます．これは,「$1, 2, ..., i$ 番目の連結成分に対して S_i, T_i のうち選ぶほうを決めて，和集合をとることでできる集合の大きさ」を動的計画法で順次求めることで計算することができます．

F: Eating Symbols Hard

$L = 10^9$ とおきます．A, P を定めたとき，X の多項式 $f_{A,P}$ を次で定めます：

$$f_{A,P} = \sum_{i=-L}^{L} A_i X^{i-P}$$

明らかに，A と A' の各要素が一致することと，$f_{A,0} \equiv f_{A',0}$ は同値です．

文字列 S に対して，高橋君が S を 1 文字目から順に食べた後に高橋君が思い浮かべている数列 A を $T(S)$ で表し，$t(S) = f_{T(S),0}$ とします．このとき，$t(S)$ に対して，次の関係が成り立つことがわかります：

- $t(\epsilon) = 0$ (ϵ は空文字列)
- $t(\texttt{+}S) = t(S) + 1$ ($\texttt{+}S$ は，文字 + と文字列 S をこの順でつなげたもの．以下同様)
- $t(\texttt{-}S) = t(S) - 1$
- $t(\texttt{>}S) = t(S)X^{-1}$
- $t(\texttt{<}S) = t(S)X$

よって，X の多項式から X の多項式への関数 $\tau_{\texttt{+}}, \tau_{\texttt{-}}, \tau_{\texttt{>}}, \tau_{\texttt{<}}$ を次で定めると，$t(S) = \tau_{S_1} \circ \tau_{S_2} \circ \cdots \circ \tau_{S_{|S|}}(0)$ となることがわかります（$\circ$ は関数合成)：

- $\tau_{\texttt{+}}(p) = p + 1$
- $\tau_{\texttt{-}}(p) = p - 1$
- $\tau_{\texttt{>}}(p) = pX^{-1}$
- $\tau_{\texttt{<}}(p) = pX$

よって，求める値は，組 (i, j) $(1 \leq i \leq j \leq |S|)$ であって，$\tau_{S_i} \circ \cdots \circ \tau_{S_j}(0) = t(S)$ なるものの個数です．i を固定したときに，これを満たす j が何個あるかを効率的に数えることを考えます．

ここで，τ_c を合成して得られる関数は，$p \mapsto Ap + B$ (A, B は X の多項式，$A \not\equiv 0$) の形をしています．$A_{[i,j)}, B_{[i,j)}$ を，それぞれ $\tau_{S_i} \circ \cdots \circ \tau_{S_{j-1}}$ を $p \mapsto Ap + B$ の形で書いたときの A, B とします．$A_{[i,j)}, B_{[i,j)}, A_{[j,k)}, B_{[j,k)}$ がわかっているとき，$A_{[i,k)}, B_{[i,k)}$ は次のようにして求められます：

$$A_{[i,k)} = A_{[i,j)} \cdot A_{[j,k)},\ B_{[i,k)} = A_{[i,j)} \cdot B_{[j,k)} + B_{[i,j)}$$

このことから，

$$B_{[i,j)} = B_{[i,k)} - A[i,j) \cdot B_{[i,k)} = A_{[i,k)} \left(\frac{B_{[i,k)}}{A_{[i,k)}} - \frac{A_{[i,j)} \cdot B_{[j,k)}}{A_{[i,k)}} \right) = A_{[i,k)} \left(\frac{B_{[i,k)}}{A_{[i,k)}} - \frac{B_{[j,k)}}{A_{[j,k)}} \right)$$

がわかります．よって，$\tau_{S_i} \circ \cdots \circ \tau_{S_{j-1}}(0) = t(S)$ のとき，

$$\frac{B_{[j,|S|+1)}}{A_{[j,|S|+1)}} = \frac{B_{[i,|S|+1)} - t(S)}{A_{[i,|S|+1)}}$$

が成り立ちます．よって，各 j に対して $\frac{B_{[j,|S|+1)}}{A_{[j,|S|+1)}}$ を計算しておけば，i を固定したとき，$j > i$ であって $\frac{B_{[j,|S|+1)}}{A_{[j,|S|+1)}} = \frac{B_{[i,|S|+1)} - t(S)}{A_{[i,|S|+1)}}$ なる j の個数を求めることで，問題の答えが得られます．

ところで，$\frac{B_{[j,|S|+1)}}{A_{[j,|S|+1)}}$ は，一般には (N 次多項式) / (N 次多項式) の有理式になるため，これをすべての j に対して求めて記録することは難しいです．一方，この有理式を直接記録する代わりに，十分多くの X および $\bmod M$ で有理式の値を計算しておくことは可能です．

十分多くの X, M に対して有理式の値が一致すれば，十分高い確率で正しい答えが得られることを示します．この問題で考える有理式は，高々 (N 次多項式) / (N 次多項式) であるため，$P/Q = P'/Q'$ すなわち $PQ' = QP'$ かの判定は，高々 $2N$ 次の多項式同士の比較に相当します．$PQ' \neq QP'$ であるにもかかわらず $(PQ')(X) = (QP')(X) \pmod M$ となるような X は，高々 $2N$ 個しか存在しません．よって，X を一様ランダムに選んでくると，高々確率 $\frac{2N}{M}$ でこの 2 つは一致していると誤って判定されます．

比較すべき有理式の組は $\frac{(N+1)N}{2}$ 組です．よって，ランダムに X を k 個とって比較する場合，いずれかの組で誤った判定が起きる確率は，高々 $\frac{(N+1)N}{2} \cdot \left(\frac{2N}{M}\right)^k$ です．$M \sim 10^9$ 程度の素数をとれば，例えば $k = 6$ でこの確率は 10^{-12} 未満となり，十分高確率で正解することができます．

このアルゴリズムの計算量は，i ごとに一致する j の発見を `map` などを用いて行うと，$O(kN \log N)$ となります．

AtCoder Beginner Contest 101 / AtCoder Regular Contest 099 Editorial

writer: semiexp

June 23rd, 2018

A: Eating Symbols Easy

- C++ example: `https://abc101.contest.atcoder.jp/submissions/2711190`
- Ruby example: `https://abc101.contest.atcoder.jp/submissions/2711191`
- Python example: `https://abc101.contest.atcoder.jp/submissions/2711198`

B: Digit Sums Easy

- C++ example: `https://abc101.contest.atcoder.jp/submissions/2711205`
- Ruby example: `https://abc101.contest.atcoder.jp/submissions/2711210`
- Python example: `https://abc101.contest.atcoder.jp/submissions/2711212`

C: Minimization

The minimum of the entire sequence is 1. Since this value never changes, we want to make all elements equal to 1.

Initially, the sequence contains only one 1. In each step, the number of 1s increases by at most $K-1$ (because the minimum of the chosen interval doesnt't change). Thus, we need at least $\lceil \frac{N-1}{K-1} \rceil$ steps.

On the other hand, we can always achieve the goal within this number of steps, as follows. Let I_c be the interval $c(K-1)+1, c(K-1)+2, \ldots, c(K-1)+K$.

Note that for each c, I_c and I_{c+1} shares an element, and $I_1, \ldots, I_{\lceil \frac{N-1}{K-1} - 1 \rceil}$ covers the entire sequence. (Strictly speaking, to avoid the intervals going outside the sequence, we shuold shift the last interval to fit within the sequence.)

Suppose that initially, I_g contains the only 1. Then, perform the following:

- For each $i = g, g-1, g-2, \ldots, 1$ in this order, perform an operation on the interval I_i.
- For each $i = g+1, g+2, \ldots, \lceil \frac{N-1}{K-1} - 1 \rceil$ in this order, perform an operation on the interval I_i.

D: Snuke Numbers

Let $f(N)$ be the integer $n \geq N$ that minimizes the value of $\frac{n}{S(n)}$ (in case of tie, choose the smallest one). We start with $N = 1$, and by repeating $N \leftarrow f(N+1)$, we can list all Snuke numbers in the increasing order. Now we want to compute $f(N)$ for a given N.

We'll prove that $f(N)$ can be always obtained by changing the last (least important) k digits of N to 9 for some k.

Let $x = f(N)$. For simplicity, let x_i denote the i-th digit of x (i.e., the digit with coefficient 10^i). Assume that $x > N$ and the d-th digit is the first (most significant) digit where x and N differ. (Note that N and x always have the same number of digits - it's easy to check that 999...999 is always a Snuke Number.) We want to prove that $x_0 = \cdots = x_d = 9$.

Suppose that for some $i < d$, $x_i < 9$. Let y be the integer obtained by changing x_i to 9 and x_d to $x_d - 1$. Clearly, $N \leq y < x$ and $S(y) \geq S(x)$ holds. This contradicts the minimality of $\frac{x}{S(x)}$. Thus, $x_0 = \cdots = x_{d-1} = 9$.

Now, we know that $x_0 = \cdots = x_{d-1} = 9$, x_d is unknown, and $x_{d+1}, \ldots$ are the same as corresponding digits of N. Let z be the value of x in case $x_d = 0$, and let $w = x_d$. Then, we get $x = z + w \cdot 10^d$, $S(x) = S(z) + w$, and $\frac{x}{S(x)} = \frac{z+w\cdot 10^d}{S(z)+w}$. Since this is a monotonous function of w, the optimal value of w is either the smallest valid value (the d-th digit of N) or the largest valid value (9). We know that d-th digit is the digit where x and N differ, thus, $x_d = 9$. (This observation is not necessary - even without this observation, the number of candidates is limited enough.)

Now we have limited candidates for x. For each candidate compute the value of $\frac{x}{S(x)}$, and $f(N)$ is the one that minimizes this value.

E: Independence

By taking the complement of the graph, we can restate the condition of the problem as follows:

> Color the vertices of the graph with two colors. Between two vertices of the same color, there must not be an edge (of the complement graph).

From now on, simply the "graph" means "complement graph".

The condition above is equivalent to the standard rule of coloring: for each edge, the color of its two endpoints must be different. Clearly, if the given graph is not bipartite, the answer is -1.

Otherwise, for each connected component, we can uniquely determine a valid coloring (except for swapping two colors). Let S_i, T_i be the set of vertices of each color in the i-th component. Then, the set of vertices in State Taka must be formed in the following way: for each i, choose one of S_i or T_i, and take their union.

"The number of roads whose endpoint cities belong to the same state" only depends on the size of S. Since we can list all possible sizes of S by a simple DP, we can solve the problem.

F: Eating Symbols Hard

Let's use rolling hashes. Fix a large prime M, and also fix a random integer X. For a string (of symbols) S, we define $t(S)$ as follows. Let A be the sequence we get when we eat S. Then,

$$t(S) = (\sum_i A_i X^i) \pmod M$$

We can use the following properties to compute $t(S)$ (note that all equations will be evaluated modulo M):

- $t(\epsilon) = 0$ (ϵ is an empty string)
- $t(\texttt{+}S) = t(S) + 1$ ($\texttt{+}S$ is a concatenation of a character + and a string S in this order)
- $t(\texttt{-}S) = t(S) - 1$
- $t(\texttt{>}S) = t(S)X^{-1}$
- $t(\texttt{<}S) = t(S)X$

Thus, $t(S) = \tau_{S_1} \circ \tau_{S_2} \circ \cdots \circ \tau_{S_{|S|}}(0)$ （$\circ$ are compositions of functions）, where functions $\tau_{\texttt{+}}, \tau_{\texttt{-}}, \tau_{\texttt{>}}, \tau_{\texttt{<}}$ are defined as follows:

- $\tau_{\texttt{+}}(p) = p + 1$
- $\tau_{\texttt{-}}(p) = p - 1$
- $\tau_{\texttt{>}}(p) = pX^{-1}$
- $\tau_{\texttt{<}}(p) = pX$

Let c be the hash value we get when we execute the entire sequence. We want to count the number of pairs (i, j) such that $\tau_{S_i} \circ \cdots \circ \tau_{S_j}(0) = c$.

Notice that the functions $\tau_{\texttt{+}}, \tau_{\texttt{-}}, \tau_{\texttt{>}}, \tau_{\texttt{<}}$ are bijective, and we can define their inverses. We have the following:

$$\begin{aligned} & \tau_{S_i} \circ \cdots \circ \tau_{S_j}(0) = c \\ \Leftrightarrow \quad & \tau_{S_N}^{-1} \circ \cdots \circ \tau_{S_i}^{-1} \circ \tau_{S_i} \circ \cdots \circ \tau_{S_j}(0) = \tau_{S_N}^{-1} \circ \cdots \circ \tau_{S_i}^{-1}(c) \\ \Leftrightarrow \quad & \tau_{S_N}^{-1} \circ \cdots \circ \tau_{S_{j+1}}^{-1}(0) = \tau_{S_N}^{-1} \circ \cdots \circ \tau_{S_i}^{-1}(c) \end{aligned}$$

For an integer i, define a_i, b_i as follows:

- $a_i = \tau_{S_N}^{-1} \circ \cdots \circ \tau_{S_i}^{-1}(0)$
- $b_i = \tau_{S_N}^{-1} \circ \cdots \circ \tau_{S_i}^{-1}(c)$

Notice that the inverses of $\tau_{\texttt{+}}, \tau_{\texttt{-}}, \tau_{\texttt{>}}, \tau_{\texttt{<}}$ are linear functions, and their compositions are also linear functions. Thus, by keeping $\tau_{S_N}^{-1} \circ \cdots \circ \tau_{S_i}$ as a linear function, we can compute a_i, b_i for all i in linear time.

Now our task is to count the number of pairs (i, j) such that $a_{j+1} = b_i$. It can be easily done with maps, and works in $O(N \log N)$ time.

Let's estimate the collision probability of hashes. If we regard $t(S)$ as a function of X, $S = S'$ if and only if $t(S) = t(S')$. Otherwise, $t(S)$ and $t(S')$ are different as functions, and since they are polynomials of degree $2N$ (after some scaling), the number of X that makes $t(S) = t(S')$ is at most $2N$. By choosing X uniformly at random from the field F_M, the probability of collision is $\frac{2N}{M}$. By using k different values for X, since we compare $\frac{(N+1)N}{2}$ pairs of polynomials, the total failure probability is at most $\frac{(N+1)N}{2} \cdot \left(\frac{2N}{M}\right)^k$. This is less than 10^{-12} for $M \sim 10^9$ and $k = 6$.