ABC 044/ARC 060 解説

1　A: 高橋君とホテルイージー

$N \leq K$ の場合，一泊あたり X 円で N 泊宿泊することになるため，答えは NX となる．$N > K$ の場合，最初の K 泊については一泊あたり X 円，残りの $N-K$ 泊については一泊あたり Y 円となるため，答えは $KX + (N-K)Y$ となる．

この問題においては，整数の入出力，四則演算等の基本的な演算，条件分岐が必要となる．

2　B: 美しい文字列

各英小文字 (‘a’–‘z’) ごとに，以下を行えばよい．

1. w 中における出現回数を数える．
2. 出現回数が奇数か偶数かを判定する．

出現回数が奇数の文字が 1 種類でもあった場合は “No” を，それ以外の場合は “Yes” を出力する．

この問題においては，ループ，文字列の入力および長さの取得などが必要となる．なお，言語によっては，入力した文字列の末尾に改行文字が含まれる可能性がある．特に，長さを取得する際に問題となりうるため，場合によっては末尾の改行を除去した方が良いこともある．普段使用している言語で改行がどのような取り扱いとなっているか，確認しておくことが望ましい．

また，多くの言語では，文字は整数として表現されているため，文字と整数は相互に変換できる (https://ja.wikipedia.org/wiki/ASCII 等を参照)．これを用いると実装が楽になるかもしれない．

3　C: 高橋君とカード

以下，$X = \max\{x_1, \ldots, x_N, A\}$ とおく．

3.1　部分点解法

部分点制約では N が小さいので，すべてのカードの選び方 ($2^N - 1$ 通り) を試すことができる．全通り試すにはビット演算を用いるのが楽である．具体的には ABC 014 B 問題 (http://abc014.contest.atcoder.jp/tasks/abc014_2) などを参照するとよい．計算量は全体で $O(N \cdot 2^N)$ となる．

3.2　満点解法 1

以下の 3 次元配列 $\mathrm{dp}[j][k][s]$ (ただし $0 \leq j \leq N$, $0 \leq k \leq N$, $0 \leq s \leq NX$) を考える．

$$\mathrm{dp}[j][k][s] = (x_1, \ldots, x_j \text{ から } k \text{ 枚選んで } x_i \text{ の合計を } s \text{ にするような選び方の総数})$$

この配列は次のように計算することができる．

$$\mathrm{dp}[j][k][s] = \begin{cases} 1 & (j = 0,\ k = 0,\ s = 0 \text{ のとき}) \\ \mathrm{dp}[j-1][k][s] & (j \geq 1,\ s < x_j \text{ のとき}) \\ \mathrm{dp}[j-1][k][s] + \mathrm{dp}[j-1][k-1][s-x_j] & (j \geq 1,\ k \geq 1,\ s \geq x_j \text{ のとき}) \\ 0 & (\text{上記いずれでもないとき}) \end{cases}$$

さて，選んだカードの x_i の平均を A にするには，各 k $(1 \leq k \leq N)$ について，k 枚選んだときの合計が kA であればよい．すなわち $\sum_{k=1}^{N} \mathrm{dp}[N][k][kN]$ が答えとなる．

動的計画法を用いることで，計算量は全体で $O(N^3 X)$ 時間となる．

3.3　満点解法 2

「選んだカードの x_i の平均が A」であることは，「選んだカードの $(x_i - A)$ の合計が 0」と言い換えることができる．すなわち，各カード i について $y_i = x_i - A$ と定義すると，選んだカードの y_i の合計が 0 となるようなカードの選び方の総数を求めればよいことになり，選んだ枚数を考慮する必要がなくなる．この考え方に基づく，$O(N^2 X)$ 時間の解法も存在する．

実装においては，y_i が負となりうることを考慮し，以下の 2 次元配列 $\mathrm{dp}[j][t]$ (ただし $0 \leq j \leq N$, $0 \leq t \leq 2NX$) を考えるとよい．

$$\mathrm{dp}[j][t] = (y_1, \ldots, y_j \text{ から } 0 \text{ 枚以上選んで } y_i \text{ の合計を } t - NX \text{ にするような選び方の総数})$$

この配列は次のように計算することができる．

$$\mathrm{dp}[j][t] = \begin{cases} 1 & (j = 0,\ t = NX \text{ のとき}) \\ \mathrm{dp}[j-1][t] & (j \geq 1,\ \text{かつ},\ t - y_j < 0 \text{ または } t - y_j > 2NX \text{ のとき}) \\ \mathrm{dp}[j-1][t] + \mathrm{dp}[j-1][t-y_j] & (j \geq 1,\ 0 \leq t - y_j \leq 2NX \text{ のとき}) \\ 0 & (\text{上記いずれでもないとき}) \end{cases}$$

1 枚も選ばないパターンを除外する必要があるため，答えは $\mathrm{dp}[N][NX] - 1$ となる．

4 D: 桁和

明らかに，$s = n$ の場合は $n + 1$ が答えである．

それ以外の場合，はじめに $2 \leq b \leq \sqrt{n}$ および $f(b, n) = s$ を満たす整数 b が存在するかどうか全探索する．もしもそのような b が存在するならば，その最小値が答えである．

そうでない場合には，$\sqrt{n} < b \leq n$ および $f(b, n) = s$ を満たす整数 b が存在するかどうか検証する必要がある．ここで，$b > \sqrt{n}$ である場合には，n は b 進表記で 2 桁となることに着目する．すなわち，上位桁を p，下位桁を q $(1 \leq q < b, 0 \leq q < b)$ とすると，

$$n = pb + q \tag{1}$$

と書くことができる．また，題意より

$$p + q = s \tag{2}$$

が成立する．ところで $b > \sqrt{n}$ であるから，式 1 より $n = pb + q \geq pb > p^2$ となり，$p < \sqrt{n}$ が導かれる．そこで，上位桁 p を全探索すればよい．式 1,2 より $b = (n - s)/p + 1$ となるから，p から b は一意に定まる．最後に，そのような b について，実際に $f(b, n) = s$ を満たすか確認すればよい．

計算量は全体で $O(\sqrt{n})$ 時間となり，満点が得られる．

5 E: 高橋君とホテル

以下の 2 次元配列 r を考える．

$$r[k][i] = (i \text{ 番目のホテルから } 2^k \text{ 日以内に到達可能な最右のホテル番号})$$

$r[0][i]$ は，$x_j \leq x_i + L$ を満たす最大の j であり，これは $x_1, \ldots, x_N$ に対し二分探索などを用いることで，効率よく求めることができる．さらに，$r[k+1][i] = r[k][r[k][i]]$ となるから，ダブリングにより $r[1][\cdot], r[2][\cdot], \ldots$ を順次求めることができる．各クエリでは，この配列 r を用いて二分探索を行えばよい．計算量は全体で $O((N + Q) \log N)$ となる．

6　F: 最良表現

x を文字列，p を正整数とする．$0 \leq i < |x| - p$ を満たす任意の整数 i に対し $x[i] = x[i+p]$ を満たすならば，p を x の周期と呼ぶ．また，x の周期の最小値を $\mathrm{per}(x)$ と表記する．例えば $\mathrm{per}(\texttt{abcabcabcab}) = 3$ である．

入力文字列 w について，その長さを N とする．以下のとおりの場合分けを行う．

(a) w が良い文字列である場合 (例: $w = \texttt{ababa}$)
(b) $\mathrm{per}(w) = 1$ の場合 (例: $w = \texttt{aaaaa}$)
(c) それ以外の場合 (例: $w = \texttt{abcabcabc}$)

(a) の場合，明らかに最良表現の項数は 1 であり，また最良表現の総数も 1 である．良い文字列かどうかの判定法については後述する．

(b) の場合，最良表現の項数は N であり，また最良表現の総数も 1 である．

(c) の場合，最良表現の項数が 2 となることが証明できる (後述の定理 5 を参照)．したがって，最良表現の総数を求めるためには，次の条件をすべて満たす整数 i の総数を求めれば良いことになる．

- $1 \leq i < N$
- $w[0..i-1]$ は良い文字列である
- $w[i..N-1]$ は良い文字列である

すなわち，w の各接頭辞および各接尾辞について，良い文字列かどうか判定できれば良いことになる．以下，w の各接頭辞が良い文字列かどうか判定する方法を述べる．接尾辞については，w の逆文字列に対し同じ方法を適用すればよい．

以下の配列 $Z[0..N-1]$ を考える．

$$Z[i] = (w \text{ と } w[i..N-1] \text{ の最長共通部分列の長さ})$$

この配列は，Z algorithm [2] を用いることで $O(N)$ 時間で求められることが知られている．また，次の観察が得られる．

観察 1. p を，$1 \leq p < N$ を満たす整数とする．$k \geq 2$ および $(k-1)p \leq Z[p]$ を満たす任意の整数 k について，$w[0..kp-1]$ は $w[0..p-1]$ を k 回繰り返した文字列である．すなわち，$w[0..kp-1]$ は良い文字列ではない．

いま，$G[1..N]$ を，次の条件を満たす bool 型配列とする．

$$G[i] = \text{true} \iff w[0..i-1] \text{ は良い文字列}$$

観察 1 を用いることで，次の処理により配列 G を求めることができる．すなわち，w の各接頭辞が良い文字列であるかどうかを効率的に判定できる．

```
1 for (i = 1; i <= N; i++) { G[i] = true; }
2 for (p = 1; p < N; p++) {
3     for (k = 2; (k - 1) * p <= Z[p]; k++) {
4         G[k * p] = false;
5     }
6 }
```

上記の計算量は $O(N \log N)$ 時間となり，満点が得られる．

なお，証明は割愛するが，上記のコードを次のように変更すると，$O(N)$ 時間解法となることが保証される．

```
for (i = 1; i <= N; i++) { G[i] = true; }
for (p = 1; p < N; p++) {
    if (G[p] == false) { continue; }        // この行を追加
    for (k = 2; (k - 1) * p <= Z[p]; k++) {
        G[k * p] = false;
    }
}
```

さらに，Z algorithm を用いない，$O(N)$ 時間の別解も存在する．ある文字列 x が良い文字列でないことと，$|x|/\mathrm{per}(x)$ が 2 以上の整数であることは同値である (後述の補題 3 を参照)．すなわち，w の各接頭辞が良い文字列かどうか判定するためには，w の各接頭辞の最小周期を求めれば良いことが分かる．これは Knuth-Morris-Pratt 法 [3] を用いることで，$O(N)$ 時間で求められることが知られている．

6.1 正当性の証明

前述の解法は，(c) の場合において，最良表現の項数が 2 となることを基にしていた．ここではその証明を述べる．

定理 2 ([1,3] 等を参照)**.** 正整数 p, q が文字列 x の周期であり，かつ $p+q-\gcd(p,q) \leq |x|$ を満たすならば，$\gcd(p,q)$ もまた x の周期である．

補題 3. x を空でない文字列とする．以下の 2 つは同値である．

(i) x は良い文字列ではない．
(ii) $|x|/\mathrm{per}(x)$ は 2 以上の整数である．

証明. 良い文字列の定義より，(ii) ならば (i) であることは明らかである．以下，(i) ならば (ii) であることを示す．

x が良い文字列でない場合，$|x|/\mathrm{per}(x) \geq 2$ は定義より明らかである．次に，$|x|/\mathrm{per}(x)$ が整数となることを示す．x が良い文字列でないということは，y を k 回繰り返した文字列が x となるような，文字列 y および整数 $k \geq 2$ が存在する．ここで $p = \mathrm{per}(x)$, $q = |y|$ とおくと，$p \leq q = |x|/k \leq |x|/2$ が成立する．p および q はともに x の周期であり，かつ $p+q-\gcd(p,q) \leq |x|$ を満たすから，定理 2 により $\gcd(p,q)$ は x の周期となる．ここで $|x|/\mathrm{per}(x)$ が整数でないと仮定すると，q は p の倍数でないことになる．その場合，$\gcd(p,q) < p$ となってしまうので，$p = \mathrm{per}(x)$ が x の最小周期であることに矛盾する．よって，$|x|/\mathrm{per}(x)$ は整数である． □

補題 4. x を長さ 2 以上の文字列とする．また $m = |x|$ とする．さらに $y = x[1..m-1]$ とおく．x が良い文字列でなく，かつ $\mathrm{per}(x) \neq 1$ であるならば，y は良い文字列である．

証明. y が良い文字列でないと仮定する．$p = \mathrm{per}(x)$, $q = \mathrm{per}(y)$ とおく．補題 3 と仮定より，p は m の約数であり，q は $|y| = m-1$ の約数ということになる．一般に m と $m-1$ は互いに素なので，p と q も互いに素，すなわち $\gcd(p,q) = 1$ となる．さらに $p \leq m/2$ および $q \leq (m-1)/2$ である．ここで，p は y の周期でもある．したがって，定理 2 より，$\gcd(p,q) = 1$ は y の周期となる．よって，x の末尾 $m-1$ 文字はすべて同じ文字となってしまい，さらに $x[0] = x[p]$ より，$x[0]$ もまた同じ文字となってしまう．結局 $\mathrm{per}(x) = 1$ となり，前提に矛盾する．したがって，y が良い文字列でないという仮定は誤りであることがわかる．ゆえに y は良い文字列である． □

定理 5. 文字列 w について，w が良い文字列ではなく，かつ $\mathrm{per}(w) \neq 1$ を満たすと仮定する．このとき，w の最良表現の項数は 2 である．

証明. 長さ 1 の文字列は明らかに良い文字列である．さらに，補題 4 より，$w[1..|w|-1]$ は良い文字列であるから，結局，列 $(w[0], w[1..|w|-1])$ は w の良い表現の一つである．また，項数 1 以下の w の良い表現が存在しないことは明らかである．よって，w の最良表現の項数は 2 である． □

参考文献

[1] N. J. Fine and H. S. Wilf. Uniqueness theorems for periodic functions. *Proc. Amer. Math. Soc.*, 16:109–114, 1965.

[2] D. Gusfield. *Algorithms on Strings, Trees, and Sequences.* Cambridge University Press, 1997.

[3] D. E. Knuth, J. H. Morris, and V. R. Pratt. Fast pattern matching in strings. *SIAM Journal on Computing*, 6(2):323–350, 1977.

ARC 060 / ABC 044 Editorial

climpet

August 28th, 2016

A: Tak and Hotels (ABC Edit)

If $N \leq K$, the answer is NX. Otherwise the answer is $KX + (N-K)Y$.

B: Beautiful Strings

For each lowercase letter c from 'a' to 'z', count the number of occurrences of c in s, and check if the number is even.

C: Tak and Cards

This problem can be solved by dp.

Let $dp[i][j][k]$ be the number of ways to choose j numbers among the first i numbers such that the sum becomes k. You can update this array from smaller i, and when $i > 0$, $dp[i][j][k] = dp[i-1][j][k] + dp[i-1][j-1][k-x_i]$. (Make sure that you don't access to negative indices).

Then, the answer is the sum of $dp[N][t][At]$ for $0 \leq t \leq N$.

Exercise: the algorithm above is $O(N^4)$. Can you improve it to $O(N^3)$?

D: Digit Sum

Consider two cases: $b \leq sqrt(n)$ or $b > sqrt(n)$.

The former case is easy. Just try all possible values of b.

In the latter case, when you write n as a base-b number, the number of digits will be at most two.

If you write n as a base-b number, you get $n = pb + q$ for some $p, q < b$. From the statement, you get $p + q = s$. By comparing these two equations, you get $n - s = (pb + q) - (p + q)$, which is equivalent to $(p-1)b = n - s$.

Thus, except for the special case $n - s$, b must be a divisor of $n - s$, and again you can try all possible divisors of $n - s$ as candidates for b.

The total complexity is $O(sqrt(n))$.

E: Tak and Hotels

Assume that $a_i < b_i$ (the other case is similar).

First, for each i, we compute $right(i)$: the maximum integer that satisfies $distance(i, right(i)) \leq L$. It means that if you start the hotel i, you can reach up to the hotel $right(i)$ in a single day. This can be computed using binary search.

In this task, you want to compute the minimum k such that $right^k(a_i) \geq b_i$.

For each i and t, you need to precompute the value of $right^{2^t}(i)$. Then, you can compute the answers for queries using binary search.

The total complexity is $O((N + Q) \log N)$.

F: Best Representation

The key observation in this task is that, unless all characters in w are the same, the minimum number of elements is always 1 or 2.

Let n be the length of w. If neither of the first $n - 1$ characters of w and the last $n - 1$ characters of w are good, in other words, if both the prefix of length $n - 1$ and the suffix of length $n - 1$ are periodic, we can prove that all characters in w are identical.

The key lemma for the proof is the following:

- Let p, q are coprime integers, and let s be a string of legnth at least $p + q - 1$. If both p and q are periods of s, all characters in s are identical.

The remaining part of the proof is an exercise for readers.

Thus, if we can get all good prefixes and good suffixes of w, we can solve the task.

In order to check if a given string s is good, for each divisor d of $len(s)$, you need to check if $s[0, len(s) - d)$ and $s[d, len(s))$ are the same. You can compare two strings in $O(1)$ time after you pre-compute rolling hashes, so in total this problem can be solved in $O(nlogn)$.

Another way to compute all good prefixes/suffixes is to use Z-algorithm (again, an exercise for readers).