CODE FESTIVAL 2016 Grand Final Editorial

writer: rng_58

A: 1D Matching

$2N$ 個の点を左から順に見ていく。まだ割り当てられていないものの集合 S を用意しておく。最初 S は空であり答えは 1 である。

現在物 p を見ているとする。S に p に割り当てられる元が存在する場合、最適解ではそのうち一つを p に割り当てる必要がある (どれを選ぶかは関係ない)。この場合、答えにそのような元の個数をかけ、S から一つの元を取り除く。それ以外の場合は、p を S に加える。

B: Inscribed Bicycle

a, b, c を三辺の長さ、r を内接円の半径とする。

半径 x の円を二つ描けるかチェックすることを考える。三角形の内部の点 P は、境界からの距離が x 以上である場合円の中心となることができる。そのような点の集合は三角形をなす。この三角形は元の三角形と相似であり、$(1 - x/r)$ 倍小さい。

この小さい三角形の中から二点を選んで距離を $2x$ 以上にできる場合のみ半径 x の円を二つ描くことができるので、答えは式 $(1 - x/r) \times \max\{a, b, c\} = 2x$　をみたすような x である。

C: Cheating Nim

不正者の目的は xor を 0 にすることである。i 番目の山から石を食べると、xor を $a_i \hat{} (a_i - 1)$ 変える事ができる。この値が $1, 3, 7, \ldots, 2^{29} - 1$ のいずれかであることが重要である。

x を最初の xor の値とする。$k = 29, \ldots, 1$ についてこの順に、$a_i \hat{} (a_i - 1) = 2^k - 1$ をみたすような i が存在し x の $(k-1)$ 番の bit が 1 ならば、この山から石を食べ x を $a_i \hat{} (a_i - 1) = 2^k - 1$ 変える。最後に、x が 0 になっていなければ答えは -1 である。そうでなければ食べた石の個数が答えである。

D: Dice Game

これは次のような問題の一般化である: 二つのジェネレータが与えられる。生成したものが与えられたときどちらのジェネレータを使ったか推測せよ。

tourist 視点、Petr 視点の二つの解法がある。tourist 視点のほうが簡単に思いつくが、Petr 視点のほうが実装が簡単である。これはゼロサムゲームなので、どちらの開放でも答えが一致する。(minimax 定理)

Petr　視点 Petr がうまくやれば、tourist がどのような戦略をとっても B 以上の確率で勝つことができない、というような B の最小値を求めればよい。

Petr　の戦略は、赤を選ぶ確率 x　のみによって表せる。

tourist は Petr の戦略を知った上で行動すると考えてよいので、このとき tourist の勝率は $\max\{p_1 x, q_1(1-x)\} + \ldots + \{p_6 x, q_6(1-x)\}$　である。B の値は　x が 0 以上 1 以下を取るときのこの値の最小値である。たとえば、三分探索により、簡単に求めることができる。

E: Water Distribution

N 個の頂点からなり、操作を行った都市の間に辺をはったグラフを考える。このグラフで、連結成分ごとに独立に考えることができる。

連結成分 C を固定する。A, B, K をこの連結成分内の元の水の量の和、辺の長さの和、頂点の個数とする。操作後の水の量の合計は $B - A$ となるので、この連結成分内の最小値は $(B - A)/K$ 以下である。

逆に、つねに $(B - A)/K$ を達成できることが証明できる。この連結成分は気であるとしてよい。葉を一つ取り、この葉の水の量が $(B - A)/K$ より大きいかどうかに応じて操作をすることで目的を達成できる。

したがって、すべての空で無い部分集合について、最小全域木を求め、次に $O(3^n)$ の DP をすればよい。

F: Intervals

N を奇数とする。 N が偶数の場合もほとんど同じである。

まず、区間の順番をすでに決めたとする。$(N+1)/2$　番目の区間を ”center”　と呼び、その左右に置かれる区間を ”left”, ”right” と呼ぶことにする。最適解では center はもとの位置のまま動かないことが証明できる。

この場合、 ”left” の区間には R_i のコスト、”right” の区間には L_i　のコストがかかる。さらに、中央に近い区間が他の区間を押しのけることによりコストがかかる。

$p_0, \ldots, p_{k-1}$ $(k = (N-1)/2)$ を左の区間のインデックス、 $q_0, \ldots, q_{k-1}$ を右の区間のインデックスとする。また、r を center の区間のインデックスとする。このとき、全体残すとは

$$\sum_{i=0}^{k-1} R_{p_i} + \sum_{i=0}^{k-1} L_{q_i} + \sum_{i=0}^{k-1} iX_{p_i} + \sum_{i=0}^{k-1} iX_{q_i} + (N-1)X_r$$

と書ける (X_i は区間 i　の長さである)

このとき、 p と q はソートされているべきであることが分かる。まず全ての区間を長さによってソートしておく。$dp[i][j][k]$ を最初の $i+j+k$ この区間を i 個を左、 j 個を右、 k (0 or 1) 個を中央に置く場合の最小コストとする。これにより $O(N^2)$ の回答が得られる。

G: FESTIVAL

次のような構成方法でできる：

”$FESTIVAL\ldots LFESTIVAL\ldots L\ \cdots\ FESTIVAL\ldots L$”

”FESTIVA” は 600 回現れ、 i 番目の ”FESTIVA” の後には c_i 回の ”L” が現れる。

”FESTIVA” を k 回繰り返した文字列に現れる文字列 ”FESTIVA” の個数を $f(k)$ とする。このとき、上で構成した文字列に現れる ”FESTIVAL” の個数は $c_1f(1)+c_2f(2)+\ldots+c_{600}f(600)$ である。$c_{600},\ldots,c_1$ を greedy にこの順に決めていくと、 5000 文字以下の文字列を必ず構成することができる。

H: AB=C Problem

行列 A と C が与えられたとき、$AB = C$ をみたす B の個数を数えることを考える。行列の定義より、 C の行ベクトルは A の行ベクトルの線形結合として表せ、 B の要素がその係数を指定している。$A_1, \ldots, A_N$ を A の行ベクトル、 $C_1, \ldots, C_N$ を C の行ベクトルとする。ある i について C_i が $A_1, \ldots, A_N$ によって生成されるベクトル空間に含まれていない場合、そのような B は存在しない。そうでなければ、そのような B の個数は $2^{N(N-R)}$ である (R は A のランクである)

すべての i について、

- 行列のランクは i である
- A によって生成されるベクトル空間が全ての C の行ベクトルを含む

これは $O(N^3)$ の DP でとくことができる。 $dp[i][j][k]$ を長さ N のベクトルの i 個組 $(v_1, \ldots, v_N)$ であって

- $(v_1, \ldots, v_N)$ によって生成されるベクトル空間のランクが j
- そのベクトル空間と、 C の共通部分のランクが k

であるようなものの個数とする。漸化式の詳細は読者に任せる。

この問題の答えは、 N と行列 C のランクのみに依存する。

I: 90 and 270

外角の和が 360 度でなければ、解は存在しない。そうでなければ必ず解が存在することを証明することができる。

$\{90, 90, 90, 90\}$ のケースを除き、 90 と 270　が隣接しているような場所が存在する。一般性を失わずに、 $a_2 = 90$ と $a_3 = 270$　であるとしてよい。

内角が $(a_1, \ldots, a_N)$ $(a_2 = 90$ と $a_3 = 270)$　である多角形を構成するために、 まずは内角が $(a_1, a_4, \ldots, a_N)$　であるような多角形を構成する。それを Q　とする。また、一般性を失わずに、 Q_2 は Q_1　の右側にあるとしてよい。このとき、P は Q　を用いて以下のように構成できる。

- $P_1 = P_2 = P_3 = Q_1$ とし、 $i \geq 2$　に対し　$P_{i+2} = Q_i$　とする。
- P_2 と P_3 を右に 0.5　ずらす。
- P_3　と P_4 を上に 0.5　ずらす。
- 全ての座標が整数となるように座標を付け直す。

この操作を再帰的に繰り返すと、 $O(N^2)$ で問題を解くことができる。

J: 123 Pairs

最初に、ペアをグループに分けることを考える。整数 x に対し、$x+0.5$ を通るペアが存在しない場合、x+0.5 の左右でグループに分割する。たとえば、ペアが $(1,3),(2,4),(5,6),(7,10),(8,9)$ のとき、グループは $\{1,3),(2,4)\},\{(5,6)\},\{(7,10),(8,9)\}$ となる。

全てのペアで差が 3 以下で無ければならないとき、グループにどのようなパターンが現れるか考える。手でパズルのようなことをすると、パターンは非常に限られていることが分かる。以下が全てのパターンである：

- Pattern 1: $(1,2)$.
- Pattern 22: $(1,3),(2,4)$.
- Pattern 232: $(1,3),(2,5),(4,6)$.
- Pattern 2332: $(1,3),(2,5),(4,7),(6,8)$.
- つづく
- Pattern 31: $(1,4),(2,3)$.
- Pattern 333: $(1,4),(2,5),(3,6)$.

三つのパターンと一つのパターンの列 $(22,232,2332,\ldots)$ がある。

差が $1,2,3$ であるようなペアの個数が与えられたとき、Pattern 1 の個数を知ることができる。p を Pattern 31 の個数とし、q を Pattern 333 の個数とする。p と q を固定すると、そのようにペアを分ける方法の個数を簡単な計算により $O(1)$ で求めることができる。したがって、全体の問題を $O(N^2)$ 出とくことができる。

参考として、FFT を用いるとこの問題は $O(NlogN)$ 時間で解くこともできる。

CODE FESTIVAL 2016 Grand Final Editorial

writer: rng_58

A: 1D Matching

We handle the $2N$ points from left to right. We keep a set of "waiting objects" S during the process. Initially S is empty and the answer is 1.

Suppose that currently we handle an object p. If there are one or more elements in S that can be matched with p, in the optimal matching, we must choose one of them and match it with p. It doesn't matter which of them to choose. Thus, multiply the answer by the number of such objects, and then removes one of such objects from S. Otherwise, add p to the set S.

B: Inscribed Bicycle

Let a, b, c be the side lengths and r be the radius of the incircle.

Let's check if we can draw two circles of radius x inside the given triangle. A point P inside the triangle can be a center of a circle if the distance between P and the boundary is at least x. The set of such Ps form a triangle region - this triangle is similar to the original triangle and is $(1 - x/r)$ times smaller.

We can draw two circles of radius x if we can put two points inside this region whose distance is at least $2x$. Thus, the answer is the x that satisfies the equation $(1-x/r)\times\max\{a, b, c\} = 2x$.

C: Cheating Nim

The objective of the cheater is to make the xor zero. If he eats a stone from the i-th pile, it changes the xor by $a_i \hat{} (a_i - 1)$. The key observation is that this is always one of $1, 3, 7, \ldots, 2^{29} - 1$.

Let x be the initial xor. Then, for each $k = 29, \ldots, 1$ in this order, if there is a pile such that $a_i \hat{} (a_i - 1) = 2^k - 1$ and currently the $(k-1)$-th bit of x is 1, eats a stone from this pile and change x by $a_i \hat{} (a_i - 1)$. After the process, if x is not zero, the answer is -1. Otherwise the answer is the number of stones we have eaten.

D: Dice Game

This task is a generalization of tasks of the following type: *There are two generators of something. You are given a generated thing. Guess which generator was used.*

There are two solutions: tourist's perspective and Petr's perspective. tourist's perspective is easier to come up with, while Petr's perspective is easier to code. Anyway, both will give the same result because this is a zero-sum game (minimax theorem).

tourist's perspective We want to find the maximum A such that if tourist follows a certain strategy, he can win with probability at least A no matter what Petr does.

tourist's strategy can be represented as a tuple of 6 numbers $(r_1, \ldots, r_6)$. It means that if he is told the number k, he answers "red" with probability r_k. We assume that Petr follows the best strategy against this strategy. In other words, Petr behaves as if he knows the values r_k.

If Petr chooses red, he loses with probability $p_1 r_1 + \ldots + p_6 r_6$. If Petr chooses blue, he loses with probability $q_1(1 - r_1) + \ldots + q_6(1 - r_6)$. Thus, tourist's winning probability (= Petr's losing probability) is $\min\{p_1 r_1 + \ldots + p_6 r_6, q_1(1 - r_1) + \ldots + q_6(1 - r_6)\}$. The value of A is the maximum of this value when $r_1, \ldots, r_6$ moves under the constraints that $0 \leq r_k \leq 1$.

Let's plot a point $(p_1 r_1 + \ldots + p_6 r_6, q_1(1 - r_1) + \ldots + q_6(1 - r_6))$ on a plane. The region where the point can be plotted can be represented as the Minkowski sum of six segments. Thus, the answer can be computed as the intesection of a diagonal $x = y$ and the Minkowski sum.

Petr's perspective We want to find the minimum B such that if Petr follows a certain strategy, tourist can win with probability at most B no matter what tourist does.

Petr's strategy can be represented as a single number x. It means that he chooses red with probability x.

Again, we treat as if tourist knows the value of x, thus tourist's winning probabiltiy is $\max\{p_1 x, q_1(1 - x)\} + \ldots + \{p_6 x, q_6(1 - x)\}$. The value of B is the minimum of this value when x moves between 0 and 1. It can be easily computed, for example using tenary search.

And again, from minimax theorem, $A = B$.

E: Water Distribution

Construct a graph with N vertices. There is an edge between the vertices i and j if the operation is performed between the cities i and j. Since there is no interaction between different connected components in this graph, we can handle each component separately.

Let's fix a connected component C. Let A be the sum of initial amount of water in this component. Let B be the total length of the edges in this component. Let K be the number of cities in this component. Clearly, the minimum amount of water in a city in this component is at most $(B - A)/K$ because the total amount of water after the operations is $B - A$.

On the other hand, we can prove that we can achieve the value $(B - A)/K$. We can assume that the graph in this component forms a tree. Take an arbitrary leaf, x, and let y be the vertex adjacent to x. If the amount of water in x is too much, we transfer the excess to y. Otherwise, we transfer the excess in $C \setminus \{x\}$ to y first and then transfer it from y to x.

Thus, for each non-empty subset of vertices, we can get the optimal strategy when this subset forms a connected component by computing MST in $O(2^n n^2 logn)$ time. After that we can apply a standard $O(3^n)$ DP.

F: Intervals

We will assume that N is odd, but the case where N is even is almost equivalent.

Suppose that we know the order of intervals after the operations. Let's call the $(N+1)/2$-th interval in the order "center". Similarly, let's call the leftmost $(N-1)/2$ intervals "left" and call the others "right". We can prove that in the optimal solution the center interval should remain at the original position.

How can we compute the optimal cost in this case? First, for each left intervals, the rightmost end must go to the left origin, so it must move by at least R_i (where i is the index of the interval). Similarly, for each right interval, we need to move it L_i. Additionally, some intervals that are close to the origin will force other intervals to move further.

Let $p_0, \ldots, p_{k-1}$ (here $k = (N-1)/2$) be the index of the left intervals from left to right, $q_0, \ldots, q_{k-1}$ be the index of the right intervals from right to left, and r be the index of the center interval. Then the cost can be written as:

$$\sum_{i=0}^{k-1} R_{p_i} + \sum_{i=0}^{k-1} L_{q_i} + \sum_{i=0}^{k-1} iX_{p_i} + \sum_{i=0}^{k-1} iX_{q_i} + (N-1)X_r$$

where X_i is the length of the interval i.

Now it is clear that both p and q should be sorted in the decreasing order of the lengths. We get the following DP. First, sort all intervals by the lengths in decreasing order. Let $dp[i][j][k]$ be the minimum cost added by the first $i+j+k$ intervals when i of them go to left, j of them go to right, and k (0 or 1) of them go to the center. This will lead to an $O(N^2)$ solution.

G: FESTIVAL

One possible answer is of the following form:

"$FESTIVAL \ldots LFESTIVAL \ldots L \quad \cdots \quad FESTIVAL \ldots L$"

Here there are 600 occurrences of "FESTIVA" and after the i-th "FESTIVA" there are c_i occurrences of "L".

How many "FESTIVAL"s does this string contain as subsequences? Let $f(k)$ be the number of "FESTIVA"s as subsequences in the k-times repetition of "FESTIVA". Then, the number of "FESTIVAL"s can be written as $c_1 f(1) + c_2 f(2) + \ldots + c_{600} f(600)$. If we decide the values $c_{600}, \ldots, c_1$ greedily in this order, we will always end up with a string with at most 5000 characters.

H: AB=C Problem

First, suppose that the two matrices A and C are given and you want to count the number of matrices B that satisfy $AB = C$. From the definition of matrix multiplication, each column vector of C is a linear combination of column vectors of A, and the elements of B specify the coefficients. Let $A_1, \ldots, A_N$ be the column vectors of A, and $C_1, \ldots, C_N$ be the column vectors of C. Then, if for at least one i, C_i is not in the vector space generated by $A_1, \ldots, A_N$, there is no B that satisfies the equation. Otherwise, the number of such Bs is $2^{N(N-R)}$, where R is the rank of the matrix A.

Thus, for each i, it is sufficient to count the number of matrices A such that

- The rank of the matrix is i.
- The vector space generated by A contains all column vectors of C.

In order to do this, we do an $O(N^3)$ DP. Define $dp[i][j][k]$ as the number of $i-tuples$ of vectors of length N, $(v_1, \ldots, v_N)$, such that

- The rank generated by $(v_1, \ldots, v_N)$ is j.
- The rank of the intersection of the vector space generated by $(v_1, \ldots, v_N)$ and the vector space generated by all column vectors of C is k.

The exact recurrence formula is left as an exercise. It is notable that the recurrence formula (and thus the answer too) only depends on N and the rank of C.

I: 90 and 270

If the sum of outer angles is not 360 degrees, the answer doens't exist. Otherwise we can prove that the solution always exists.

Except for the trivial case $\{90, 90, 90, 90\}$, there is an adjacent pair of 90 and 270. Without loss of generality, we can assume that $a_2 = 90$ and $a_3 = 270$.

In order to construct a polygon P with inner angles $(a_1, \ldots, a_N)$ (where $a_2 = 90$ and $a_3 = 270$), we first construct a polygon for $(a_1, a_4, \ldots, a_N)$. Let's call it Q. Also, without loss of generality, we can assume that Q_2 is to the right of Q_1. Then, P is defined from Q as follows.

- Let $P_1 = P_2 = P_3 = Q_1$ and $P_{i+2} = Q_i$ for each $i \geq 2$.
- Shift P_2 and P_3 to the right by 0.5.
- Shift P_3 and P_4 upward by 0.5.
- Renumber coordinates to make all coordinates integers.

If we apply this process recursively, we can get the desired polygon in $O(N^2)$ time.

J: 123 Pairs

First, for a given set of pairs, we divide the pairs into groups. For some integer x, if there is no pair that passes $x + 0.5$, the pairs to the left of $x + 0.5$ and the pairs to the right of it belong to different groups. For example, when the pairs are $(1, 3), (2, 4), (5, 6), (7, 10), (8, 9)$, the groups are $\{1, 3), (2, 4)\}, \{(5, 6)\}, \{(7, 10), (8, 9)\}$.

Let's think about the "patterns" of groups that can appear when the difference in each pair must be at most 3. If we do some puzzle by hand, we find that actually the number of patterns is very limited. Here is the compete list of all patterns:

- Pattern 1: $(1, 2)$.
- Pattern 22: $(1, 3), (2, 4)$.
- Pattern 232: $(1, 3), (2, 5), (4, 6)$.
- Pattern 2332: $(1, 3), (2, 5), (4, 7), (6, 8)$.
- and so on.
- Pattern 31: $(1, 4), (2, 3)$.
- Pattern 333: $(1, 4), (2, 5), (3, 6)$.

There are three patterns and a family of patterns $(22, 232, 2332, \ldots)$.

When the number of pairs with differences $1, 2, 3$ are given, we clearly know the number of groups with Pattern 1. Let p be the number of Pattern 31 and let q be the number of Pattern 333. For a fixed p and q, we can count the number of valid pairings in $O(1)$ time with simple combinatorics. Thus, the entire problem can be solved in $O(N^2)$.

Note that if we use FFT, it is also possible to get an $O(NlogN)$ solution (but we allowed $O(N^2)$ because the reduction is rather straightforward).