Problem Tutorial: "Interesting Subsegments"

Consider some array. Let's calculate prefix sums modulo 3. If there are x sums equal to 0, y sums equal to 1 and z sums equal to 2 number of interesting subsegments will be:

$\binom{x}{2} + \binom{y}{2} + \binom{z}{2}$

Also, lexicographically smallest array corresponding to such values will have form $x-1$ zeroes, 1, $y-1$ zeroes, 1, $z-1$ zeroes.

So, we are interested in finding lexicographically **largest** solution:

$\binom{x}{2} + \binom{y}{2} + \binom{z}{2} = k$

$x + y + z = n + 1$

We can iterate over values of x in decreasing order, then numbers y and z are reconstructed uniquely(by solving quadratic equation).

Problem Tutorial: "Even Forest"

Paint all vertices of G in black and white alternately. A tree in the forest will be even if all of its leaves will be of the same color. Define the *color* of an even subtree as the color of its leaves.

We can hang the tree from any vertex and use DFS to compute for each vertex v the following three values.

- $s(v)$ — minimum number of edges to be removed from the subtree below v so that the obtained subforest is even and v is in a tree of its own color.
- $o(v)$ — minimum number of edges to be removed from the subtree below v so that the obtained subforest is even and v is in a tree of opposite color.
- $p(v)$ — minimum number of edges to be removed from the subtree below v so that the obtained subforest in union with v's parent vertex and parent edge is even and v is in a subtree of opposite color (this has the effect of allowing a leaf v to temporarily violate its tree color provided that v will later be connected to its parent and thus have higher degree in future)

Note that there are cases when $o(v)$ or $p(v)$ are undefined (for example if v is a leaf). In such cases we set them to ∞.

Assume that during DFS(v) these values have been computed for all children of v. This is how we compute $s(v)$, $o(v)$, $p(v)$.

- $s(v)$. For each child u we can either keep edge $\overline{uv}$ and remove $p(u)$ edges below u or remove $\overline{uv}$ and $s(u)$ edges below u. Thus
$$s(v) = \sum_{u \in \text{children}(v)} \min\{p(u), s(u)+1\}.$$
- $o(v)$ and $p(v)$. For a node to be in a tree of opposite color it must not be a leaf. Depending on whether its parent edge will be included in its tree or not at least 1 or 2 of its children must be in trees of their own color so that it can be connected to them and have $\deg v > 1$ in the resulting forest.

  Here again for each child u we can either keep $\overline{uv}$ and remove $s(u)$ other edges or remove $\overline{uv}$ and $o(u)$ more edges. According to this, the optimal sum would be
$$\sum_{u \in \text{children}(v)} \min\{s(u), o(u)+1\}.$$
  But since we also have to keep at least 2 children connected for $o(v)$ and at least 1 for $p(v)$, we may be forced to in some cases chose $s(u)$ over $o(u)+1$ even if $s(u) > o(u)+1$. Choosing $s(u)$ over

$o(u)+1$ worsens the result by $s(u)-(o(u)+1)$, thus in this case we need to connect the children which minimize $s(u)-(o(u)+1)$.

The answer to the problem is $\min\{s(\text{root}), o(\text{root})\}$.

Problem Tutorial: "Yellow Blue Bus"

Suppose that (a, b) is center of the circle and R is radius.

For blue point (x_i, y_i) we can rewrite condition as:

$(x_i-a)^2+(y_i-b)^2 \geq R \leftrightarrow -2ax_i-2by_i-(R^2-a^2-b^2)+(x_i^2+y_i^2) \geq 0$.

For yellow point (x_i, y_i) we can rewrite condition as:

$(x_i-a)^2+(y_i-b)^2 \leq R \leftrightarrow 2ax_i+2by_i+(R^2-a^2-b^2)-(x_i^2+y_i^2) \geq 0$.

So, if we denote $R^2-a^2-b^2=c$ conditions are equivalent to point (a, b, c) belonging to the intersection of some halfspaces(of the form $(-2x_i, -2y_i, -1, x_i^2+y_i^2)$ or $(2x_i, 2y_i, 1, -x_i^2-y_i^2)$).

There exist some advanced algorithms for this problem, which can solve this problem in $O((n+m)log(n+m))$. Also, it can be solved in $O((n+m)log(eps^{-1})^2)$ using nested ternary searches `https://codeforces.com/blog/entry/61710` (Blogewoosh 8 when?).

Problem Tutorial: "Permutation Matrix"

There is no such matrix 2×2.

Construct a matrix 4×4:

$$\begin{bmatrix} 1 & 15 & 3 & 13 \\ 16 & 2 & 14 & 4 \\ 5 & 11 & 7 & 9 \\ 12 & 6 & 10 & 8 \end{bmatrix}$$

Then, using four matrix 4×4, you can construct a matrix 8×8. You can divide a smaller (4×4) matrix into the «small part» (elements ≤ 8), and «big part» (elements > 8). Determine a matrix $m(x, y)$ as a matrix 4×4, with the value x added to each «small> element, and the value y added to the «big» elements.

So we can construct a matrix:

$$\begin{bmatrix} m(0,48) & m(16,32) \\ m(32,16) & m(48,0) \end{bmatrix}$$

This is the following matrix:

$$\begin{bmatrix} 1 & 63 & 3 & 61 & 17 & 47 & 19 & 45 \\ 64 & 2 & 62 & 4 & 48 & 18 & 46 & 20 \\ 5 & 59 & 7 & 57 & 21 & 43 & 23 & 41 \\ 60 & 6 & 58 & 8 & 44 & 22 & 42 & 24 \\ 33 & 31 & 35 & 29 & 49 & 15 & 51 & 13 \\ 32 & 34 & 30 & 36 & 16 & 50 & 14 & 52 \\ 37 & 27 & 39 & 25 & 53 & 11 & 55 & 9 \\ 28 & 38 & 26 & 40 & 12 & 54 & 10 & 56 \end{bmatrix}$$

Doing so as many times as needed, you can construct a matrix for any given $n > 1$.

Problem Tutorial: "Anti-stress"

We will draw lines, such that they will divide points into two halfs(where point on a line can belong to any of the halfs).

Let's fix two perpendicular directions. Draw two lines, dividing point into halfs, parallel to this directions.

It will divide plane into 4 quadrants, let's name this 1(right-up), 2(left-up), 3(left-down), 4(right-down). Let's denote by $cnt_b(x)$ number of blue points in region x, by $cnt_y(x)$, number of yellow points in region x.

Suppose that it happened, that $cnt_b(1) = cnt_y(3)$. Then, valid answer is the following:

Choose red point as the intersection of this two lines.

1. Pair blue points in 1 region and yellow in 3 region. Each of the angles will be not acute.

2. Pair blue points in 3 region and yellow in 1 region. Each of the angles will be not acute.

3. Do similar things for 2 and 4.

For this pairing to exist, we need to proof that $cnt_b(3) = cnt_y(1)$ and similar equations for $2, 4$ regions.

Indeed, $cnt_y(3) + cnt_b(3) + cnt_y(2) + cnt_b(2) = n$, $cnt_y(1) + cnt_b(1) + cnt_y(2) + cnt_b(2) = n$. So, $cnt_y(3) + cnt_b(3) = cnt_y(1) + cnt_b(1)$, and, since $cnt_b(1) = cnt_y(3) \to cnt_b(3) = cnt_y(1)$.

$cnt_y(2) - cnt_b(4) = n - cnt_y(1) - cnt_b(1) - cnt_b(2) - cnt_b(4) = n - cnt_y(1) - (n - cnt_b(3)) = cnt_b(3) - cnt_y(1) = 0$.

From similar arguments, $cnt_y(4) = cnt_b(2)$.

We will actually proof, that there always exists direction such that $cnt_b(1) = cnt_y(3)$.

For angle α let's denote by $f(\alpha)$ difference $cnt_b(1) - cnt_y(3)$ if we consider lines in the directions α and $\alpha + \frac{\pi}{2}$(note that order of lines actually matters, since some regions will be swapped). Actually, this value is not defined properly, because there can be points belonging to this two lines and we can divide them into halfs in different ways. So instead, we will consider range of values that this function can atain. This range will be contigous integer range(you can note that after swapping any two points answer is changed by at most 1). Then, we can note that if $x \in f(\alpha)$ then $-x \in f(\alpha + \frac{\pi}{2})$. Also, if we will rotate our line answer will be changed discretly(i.e. we will not skip any value). This allows us to do binary search for direction(similarly to the way we find roots of continious function if we have two numbers at which values of function have different signs).

Also, you should be careful about case where there is point in the intersection of two lines(in this case he can belong to any of the 4 regions).

Problem Tutorial: "Mismatch"

For each value of x let's calculate how many numbers contain x as submask, let's denote this value by a_x. Then, by inclusion-exclusion formula, answer will be equal to:

$\sum_{x=0}^{x=2^{19}-1} (-1)^{popcount(x)} \binom{a_x}{k}$, where $popcount(x)$ is equal to number of bits in binary representation of x.

This can be optimized using fft, by grouping a_x by the value of x(i.e. we can consider $f_t = \sum_{a_x=t} (-1)^{popcount(x)}$).

Problem Tutorial: "Lucky Tickets"

We will prove the following: the sum of luckiness over all tickets with not equal digits is equal to 0 modulo q.

Let's consider some multiset of numbers $A = a_1, \dots a_q$, such not all of them are equal. We will prove that if we calculate sum of luckiness over all permutations of $a_1, \dots a_q$ it will be equal to zero.

Consider some arbitary submultiset of this subset $B = b_1, \dots b_k$, $k > 1$. Let's calculate with which coefficient $b_1 \cdot .. b_k \cdot$ will be present in sum $(a_1 + 1)(a_2 + 2) \cdots (a_q + q)$ over all permutations of A. It will be equal to:

$N_B \cdot N_{A \setminus B} \sum_S S_1 \dots S_{q-k}$, where N_X denotex number of permutations of subset X, where sum is taken over all subsets of S of size $q - k$.

Let's calculate $S_1 \dots S_{q-k}$ over all S. It's easy to see, that it's equal to coefficent of x^k in polynomial $(x + 1)(x + 2) \dots (x + q)$.

It's well known fact(equivalent to Fermat's little theorem) that $(x+1)(x+2)\dots(x+q) = x^q - x \ mod \ q$. So, for $1 < k < q$ this sum will be equal to $0 \ mod \ q$. Let's consider $k = 1$ and $k = q$:

1. For $k = q$ total sum will $N_A a_1 a_2 \dots a_q$. If not all elements are equal than $N_A = 0$ (since it's $q!$ divided by some smaller factorials).

2. For $k = 1$, from our analysis, sum without second part will be equal to $-N_{A\setminus x}$, where $b_1 = x$. Let's calculate sum for the second part:

It will be equal to $N_{A\setminus x}(\sum_{i=0}^{q-1} 2^i) = N_{A\setminus x}(2^q - 1) = N_{A\setminus x}$.

So, overall sum is equal to 0.

So, we need to take into account only multisets of q equal numbers for which sum will be equal to $N_A a_1 \cdot a_q = N_A a_1^q = N_A a_1 = a_1$, because of Fermat's little theorem and fact that $N_A = 1$. We can iterate over all possible values of this numbers, check if they satisfy condition and add their value to the answer.

Problem Tutorial: "Diversity Street"

First let's understand how to solve problem if we can't delete any restriction.

Then, for each house we can calculate it's minimum height(by taken maximum over all restrictions, which pass through this house). Then, it's see to see, that answer will exist iff $h_1 \le 1$, $h_2 \le 2$, ..., $h_n \le n$, where h_i denotes minimum height restriction in sorted order.

We can note it's the the same as:

1. At least n values of h are not more than n.

2. At least $n-1$ values of h are not more than $n-1$.

...

At least 1 value of h is not more than 1.

(This way of formulation will help to solve general problem).

Let's get back to general problem.

For each house let's calculate it's minimum height restriction, street restriction on which it's attained and minimum height restriction if we delete this street restriction(we can do it, for example, using scanline and set).

Now, let's calculate values b_i = how many h are not more than $i - i$. Answer will exist iff $b_i \ge 0$, for all i.

How this values will be changed if we delete some restrictions? We can iterate over all houses for which this restriction was the maximum one, then some segment of values of b should be increased by 1 (from previous value of h on this position to the current one).

So, we can do this updates and check if all b_i are not less than 0.

We can do it efficiently using segment tree with lazy propagation(by maintaing minimum and checking if it's not less than 0).

After finding which restriction to delete, we can reconstruct answer greedily.

Problem Tutorial: "Disbalance"

For case of $n = 1$, $d = m + 1$ is always true after m-th minute. Let's look at $n > 1$, and denote a_i as number of bacteria in dish number i:

1) At first, each dish i has $a_i = 1$ bacteria. After each minute, a random a_i is selected with probability $\frac{a_i}{\sum_{j=1}^{n} a_j}$ to be inreased by 1. Let's prove by induction that after k minutes each of the possible combinations of a_i, where $\sum_{i=1}^{n} a_i = n + k, min(a_i) \ge 1$ are equiprobable:

With $k = 0$, there's only one possible combination. Let's denote $p(a_1, a_2, ..., a_n)$ as probability of having those numbers of bacteria in dishes after $a_1 + a_2 + ... + a_n - n$ minutes. Then let's look at distributions

after k minutes, if all possible distributions after $k-1$ minutes are equiprobable (and let's denote this probability as p_{k-1}):

From definition of this distribution, with $a_1 + a_2 + ... + a_n = n + k$ and $min(a_i) \geq 1$, $p(a_1, ..., a_n) = \sum_{i=1}^{n} \frac{a_i - 1}{n+k-1} p(...a_{i-1}, a_i - 1, a_{i+1}, ...) = \sum_{i=1}^{n} \frac{a_i - 1}{n+k-1} p_{k-1} = \frac{k}{n+k-1} p_{k-1}$. This value is independent of values of a_i.

Thus, for a fixed k, all C_{n+k-1}^{n-1} combinations of n positive integral numbers with sum $n+k$ in Petri dishes are equiprobable.

2) Let's denote mathematical expectation of sum of d after the first k minutes with n dishes as $f(n, k)$, and expectation of value of d after the k-th minute with n dishes as $g(n, k)$. Because mathematical expectation of sum is equal to the sum of mathematical expectations, and with $k < n - 1$ holds $d = 0$, $f(n, k) = \sum_{m=n-1}^{k} g(n, m)$.

Let's note, that if $a_i \leq max(a_1, ..., a_{i-1}, a_{i+1}, ..., a_n)$, then $(a_i - a_1 - ... - a_{i-1} - a_{i+1} - ... - a_n) \leq 0$. Because of this, and because probabilities of distribution of values between a_i are independent of their order, $g(n, m) = n \cdot E(max(a_1 - a_2 - ... - a_n, 0) | \sum_{i=1}^{n} a_i = n+m) = n \cdot E(max(2a_1 - n - m, 0) | \sum_{i=1}^{n} a_i = n+m)$. Let's denote $\Delta = max(2a_1 - n - m, 0)$

Let's look at two cases: with even $n + m$ and with odd $n + m$:

3) Case with even $n + m$. In this case, $m = n + 2b$, where $b \geq 0$. Then $\forall x \in [1, b + 1]$ there's exactly $C_{2n+2b-(n+b+x)-1}^{n-2} = C_{n+b-x-1}^{n-2}$ combinations of a_i such that $\Delta = 2x$. Then:

$$E(\Delta) = \frac{\sum_{x=1}^{b+1} 2x C_{n+b-x-1}^{n-2}}{C_{n+m-1}^{n-1}} = \frac{2}{C_{n+m-1}^{n-1}} \sum_{x=1}^{b+1} x C_{n+b-x-1}^{n-2}$$

$$\sum_{x=1}^{b+1} x C_{n+b-x-1}^{n-2} = \sum_{y=0}^{b} (b+1-y) C_{n-2+y}^{n-2} = \sum_{c=0}^{b} \sum_{y=0}^{c} C_{n-2+y}^{n-2} = \sum_{c=0}^{b} C_{n-1+c}^{n-1} = C_{n+b}^{n}$$

$$g(n, n+2b) = \frac{2n C_{n+b}^{n}}{C_{2n+2b-1}^{n-1}} = \frac{2(n+b)!(n+2b)!}{b!(2n+2b-1)!}$$

4) Case with odd $n + m$. In this case, $m = n + 2b - 1$, where $b \geq 0$. Then $\forall x \in [1, b + 1]$ there's exactly $C_{2n+2b-1-(n+b-1+x)-1}^{n-2} = C_{n+b-x-1}^{n-2}$ combinations of a_i such that $\Delta = 2x - 1$. Then:

$$E(\Delta) = \frac{\sum_{x=1}^{b+1} (2x-1) C_{n+b-x-1}^{n-2}}{C_{n+m-1}^{n-1}} = \left[remember\ case\ 3 \right] = \frac{2C_{n+b}^{n} - \sum_{y=0}^{b} C_{n-2+y}^{n-2}}{C_{n+m-1}^{n-1}} = \frac{2C_{n+b}^{n} - C_{n+b-1}^{n-1}}{C_{n+m-1}^{n-1}}$$

$$g(n, n+2b-1) = \frac{(n+b-1)!(n+2b)!}{b!(2n+2b-2)!}$$

5) As such, now for any $g(n, m)$ we can calculate it in $O(1)$, if we precalculated $x!$ and $(x!)^{-1}$ for x up to $n + m$. So, with $n, k \leq 10^6$, if we precalculate those factorials up to $2 \cdot 10^6$ we can calculate $f(n, k)$ in $O(max(k + 2 - n, 1))$ time.

Problem Tutorial: "Spiral Matrix"

To solve the task, you should just notice this fact: a matrix is a *spiral matrix* if there is no more than one *bad* number, and for every element on a border except for a bad number there is a neighbouring element inside the matrix. A bad number is a matrix element with value d for which there are no neighbouring cells with value $d + 1$.

So, you can just store prefix-sums to check this conditions.