Problem A. Puzzle: X-Sums Sudoku

When the side length of a Sudoku puzzle is a power of two, it is always possible to greedily fill the minimum number that does not conflict with already filled numbers in each cell in lexicographical order. And it can be found that $a_{r,c} = ((v2^m + u) \oplus (c-1)) + 1$, where $r = u2^n + v + 1$.

Denote that $f(x, g) = \sum_{i=0}^{x-1} (i \oplus g)$. For an interval $[l, l+2^k)$ satisfying where $2^k | l$, $\{i \oplus g : i \in [l, l+2^k)\}$ are suffled consecutive integers, so the sum can be calculate in $O(1)$. $[0, x)$ can be divided into such $O(\log x)$ intervals, so $f(x, g)$ can be calculate in $O(\log x)$.

X-sum can be represented by the sum of $O(1)$ fs so that it can be calculated in $O(n + m)$.

Problem B. Puzzle: Patrick's Parabox

When the box move, the player must be adjacent to the box. There are at most $4nm$ tuples (p_x, p_y, b_x, b_y) satisfying the player is adjacent to the box. Using 01BFS on these tuples is enough.

In order to determine if the player can move to another position without moving the box, a block cut tree is used, so the complexity is $O(nm \log(nm))$.

Problem C. Puzzle: Hearthstone

Consider maintain a sequence consisting of `test` x for $x = 1, 2, \cdots, n$, and some `add`s. Initially, the sequence is `test` 1, `test` 2, $\cdots$, `test` n.

For an event `add`, append an `add` to the sequence.

For an event `test` x y, if $y = 1$ and there is not `add` after `test` x in the sequence, there is a bug; otherwise, if $y = 1$, remove the earliest `add` after `test` x. And then whatever y is, move the `test` x to the end of the sequence.

After an event, there is a bug if there exists a prefix of the sequence such that the number of `add`s is greater than the number of `test`s.

The number of secret which must not exist is the number of `test`s after the last `add`. The number of secret which must exist is the number of `add`s in the longest prefix satisfying that the number of `add`s equals to the number of `test`s.

It is easy to use `std::set` to maintain the sequence and use segment tree to find the longest prefix satisfying that the number of `add`s equals to the number of `test`s. The complexity is $O(N \log N)$ where $N = n + q$.

Problem D. Poker Game: Decision

Brute force with $6! = 720$ extremely fast comparisons between hands can pass. It can be found that there are only $\binom{6}{3} = 20$ final states, so 20 comparisons are enough.

Problem E. Poker Game: Construction

It is always possible that Alice can win. If the ranks of Alice's cards are different from Bob's cards, Alice can have at least three of a kind. Otherwise, Alice can win for the different ranks of community cards.

Bob can win in most situations except:

- AA vs AXo (o means different suits), X is one of 6, 7, 8, 9;
- XX vs XX;
- XY vs XYo.

For Bob wins, flush and straight need to be considered.

For the draw, each rank should appear at times of even number. There is no draw when:

- XX vs any except XX;
- XY vs ZZ;
- XY vs ZW, where Y<X<W<Z;
- XY vs ZW, where W<Y<X<Z;
- XY vs ZW, where W<Z<Y<X.

Problem F. Longest Common Subsequence

If $s_i = t_j$, then $s[i...] = t[j...]$. So just for each number x, find the leftmost position in s and t, denoted i and j, and the answer is at least $\min(n - i + 1, m - j + 1)$.

Problem G. Lexicographic Comparison

Maintain the cycles of permutation p by using BST such as splay and treap. To compare A_x with A_y, it need to find the cycle including minimum index satisfying $x - y$ is not divisible by the length.

There are at most $\sqrt{n}$ cycles with different lengths. For each length maintain a minimum index and check each brutely. The complexity is $O(q\sqrt{n})$.

Problem H. Expression Evaluation

For each term, maintain an integer p denoting the product of non-negative integers. For each integer, let $q := p$ and $p := 0$, and then for each digit d, let $p := 10p + dq$. So it only needs constant times of add instructions for each digit.

Problem I. Equivalence in Connectivity

Use the hash values of the DSUs. It is equivalent to offline connectivity problem after changing the tree structure to the sequence structure by DFS sequence. It can be solved in $O(N \log^2 N)$ by segment trees or $O(N \log N)$ by link-cut trees, where $N = n + m + k$.

Problem J. Symmetry: Tree

If the there is only one centroid, the centroid must lie on the axis of symmetry. Let centroid be the root, if there is a pair of two same subtrees, the subtree can lie on the both sides of axis separately. For the centroid, there can be at most two unpaired subtrees. For one unpaired subtree, there can be at most one unpair subtree of it, and so on. It can be done in $O(n \log n)$.

If there are two centroids, two centroids can also lie on the both sides of axis separately.

Problem K. Symmetry: Convex

For each i, check if the perpendicular bisector of p_1p_i is an axis and if the bisector of $p_ip_1p_2$ is an axis. Execpt the two possibility, the $p_ip_1p_2$ must be symetric to an angle of C_n. Since $p_ip_1p_2$ are different for different values of i, it only need to check at most n times.

For checking, transform the polygon to a sequence of the length of sides and the angles. Use Manacher or brute force to check if it is symetric.

Problem L. Symmetry: Closure

Case 1: If two of the lines intersect with an angle of x which is not a multiple of 45 degrees.

Case 1.1: If all lines intersect at one point, $C(A)$ and $C(B)$ will be a dense subset of a circle.

Case 1.2: Otherwise $C(A)$ and $C(B)$ will be a dense subset of the whole plane.

In other cases, the plane will be divided into many symmetric regions. Only the reflected points in the same region can contribute to the answer.

Without loss of generality, we can rotate and translate the first line to the x-axis in the following cases.

Case 2: If there are no intersections between lines.

Case 2.1: If there is only one line, then the plane will only be divided into two parts.

Case 2.2: Otherwise the period of the regions will be two times the greatest common divisor of the distance between lines.

Case 3: If the lines intersect at multiples of 90 degrees.

Without loss of generality, we can translate one of the lines perpendicular to the first line to the y-axis in this case. After that, the two dimensions will be independent of each other. We can do the previous case on two dimensions separately.

Case 4: If the lines intersect at multiples of 45 degrees.

Without loss of generality, we can translate one of the lines 45 degrees to the first line to the origin in this case.

Case 4.1: If all lines intersect at one point, then the plane will be divide into eight parts.

Case 4.2: Otherwise, the final regions will be triangular grids. However, since we pick the first line randomly, so there are two types of triangular grids.

If we temporarily remove the lines perpendicular to the first line and calculate the gcd of intersections of 45 degree lines, we will obtain a triangular grid which has the hypotenuse of the right triangle lying on x-axis.

We can consider about how the horizontal and vertical lines cut the triangles. If after doing gcd the right angle of the right triangle was cut by a line, then the grid will be switched into "leg on x-axis"case.

You need to check intersections and do rotations by using only integers, otherwise there may be precision issues.

Rotating (a, b) onto x-axis can be done by "multiplying $a - bi$".