Stage 2: Division 1

Contest Analysis

Zhejiang University
Mysterious Oscar and her friends

December 2022

A. Floor Tiles in a Park

Solution 1: Case study.
Solution 2: Use brute force to find solutions to $n, m \leq 5$, and interpolate for the polynomial.

B. Rotate Sum 3

If the polygon does not contain any axes of symmetry, then output 0.
If the polygon contains 2 or more axes of symmetry, then the answer will be a ball centered at the intersection of axes, having a radius equal to the distance to the farthest point from the center.
If the polygon contains exactly one axis of symmetry, then the answer can be represented as a sum of (truncated) cones. The total volume can be calculated by enumerating all edges on one side of the axis and add their contribution to the answer.
To find the axes of symmetry, use the **dot product** of adjacent edges and the squared length of each edge to serialize the polygon. After that, hashing or Manacher's algorithm can be applied to find the axes of symmetry.

If all the A_i are equal, then no solution exists.
Otherwise for the first $n - 1$ positions, the smallest or the largest element will always be available. For the last position, if the current answer does not satisfy the constraints, you can always exchange it with some previous element.

D. The Pool

Assume that the rectangle $ABCD$ is placed on the grid, then we can translate the rectangle so that point A at the origin.
For cases such that $n = m$, point B can be in the set $\{(a, b) | a^2 + b^2 = n^2, a > 0, b \geq 0\}$. For cases such that *nneqm*, point B can be in the set $\{(a, b) | a^2 + b^2 = n^2, b > 0 \vee (b = 0 \wedge a > 0)\}$.
(Checking whether C and D are integer points is also needed)
Assume that we have found these sets, then the answer is two times the number of squares in the triangle with legs of length a, b, plus two times the number of squares in the triangle with legs of length c, d, plus $(a + c)(d - b)$. The last part is the number of squares in the central rectangle (possibly negative).

D. The Pool

Now, the problem changes into finding a, b such that $a^2 + b^2 = n^2$. This can be done by utilizing Pollard Rho and the Gaussian integers factorization algorithm. Choosing any subset of the Gaussian factors and multiply them together will result in a possible a, b pair. For $n \leq 10^{18}$, the number of distinct a, b pairs is at most 4×295245, where $n = 495229111954868525$.
After finding the possible pairs, the number of squares in a rectangle can be calculated by Picks's Theorem or Euclid-like Algorithm.

E. Ternary Search

Without loss of generality, we can assume that the final sequence is u-shaped.

We can consider the numbers from large to small. The contribution for a number is $\min(L, R)$, where L is the number of values smaller than it to the left, and R is the number of values smaller than it to the right. After each insertion, we can use a segment tree to maintain $R - L$ for each value and calculate the sum of $\min(L, R)$.

For the n-shaped case, the answer can be obtained by changing all a_i into $-a_i$ and run the previous algorithm. The final answer is the minimum result of two cases.

F. Candies

$(a, a), (a, x - a), (x - a, a), (x - a, x - a)$ are all the pairs that can be taken in one operation. If we change all a into $x - a$ if $a < x - a$, then the answer will still be the same.

Use a stack to simulate the operations. Delete the matching elements at the beginning and the ending of the array since its circular. Orders are irrelevant.

G. Regular Expression

It is not hard to find out that `.*` can match all strings.
When the target string length is 1, the answer is 2. (a,`.`)
When the target string length is greater than or equal to 2, we only need to check if the following patterns match the target string:
(`ab`,`a.`,`.b`,`..`,`a+`,`a*`,`.+`,`.*`)

H. Grammy Sorting

Use Bipolar Orientation to find the final permutation.
Find any spanning tree and convert the problem into a tree problem.
If the tree is a star, then the problem is equivalent to taking a number at a time and swap the number with any position in the sequence, which can be solved in $O(n)$.
If the tree is a chain, then the problem is equivalent to insertion sort, which can also be solved in $O(n)$.
If the tree is several chains connected together at the root, then we can merge the solutions above to solve it in $O(n)$.
This hints us to take out the chains from the leaves to the first splitting point and solve this part of the tree in each round.
After $O(\log n)$ rounds, the process will terminate.

I. Suffix Sort

Consider how to determine which of S[i:] and S[j:] is larger under the minimal representation, we can find the length of the longest common prefix of the two suffixes under the minimum representation, and then compare the following letter in the minimal representation:
Consider to do binary search on the length of the longest common prefix, then the problem turns into judging whether two strings are isomorphic under the minimal representation.

I. Suffix Sort

Solution 1: It can be observed that for a substring with a fixed left endpoint, after the original character is mapped to the minimal representation, the size relationship between characters is unchanged, that is, we can preprocess the corresponding character size relationship (the character ranked 1st in the minimal representation, the character ranked 2nd in the minimal representation ...) for each suffix S[i:]. We can enumerate the character ranked kth of S[i:] and S[j:] according to the mapped order, and to find the situation where only the kth character is considered, S[i:] and S[j:] The position of the first mismatch. Suppose that the k-th character that appears in S[i:] appears at the positions $\{a_1, a_2, \ldots, a_p, \ldots\}$ and the positions where the kth character appears in S[j:] are $\{b_1, b_2, \ldots, b_p, \ldots\}$, then the necessary and sufficient condition of "the pth occurrence of the kth character is not mismatched" is $a_1 - i = b_1 - j, a_2 - a_1 = b_2 - b_1, a_3 - a_2 = b_3 - b_2, \ldots, a_p - a_{p-1} = b_p - b_{p-1}$.

It is not hard to discover that except the first position, the array is the difference array of the occurrence positions. We can use Suffix Array to preprocess the longest common prefix of all the suffix of the difference arrays and query them in $O(1)$. Total time complexity is $O(26n \log n)$

Solution 2: Use hashing and binary search to compare the suffixes of the difference array. We only need to change the first occurrence of each letter each time when we move a character. Total time complexity is $O(26n\log n + n\log^2 n)$

J. Melborp Lacissalc

Consider the reversed problem, we can use a prefix sum(mod k) to calculate t when the array is given.
Since the prefix sum array has a bijection to the original array, we can calculate the number of prefix sum arrays.
$f(i, j, k)$ represents the number of prefix sum arrays after placing values 0 i, using j positions, and having a goodness of k. Then the answer to the problem is $f(k-1, n, t)$. Total time complexity is $O(n^5)$.

K. Great Party

If the number of stone piles is odd, then the first player will win. Otherwise if the xor sum of (the number of stones in each pile minus one) is zero, then the first player will win. Otherwise the second player will win. The remaining part can be done by preprocessing the prefix xor sum array and using Mo's Algorithm.

L. Maximum Range

For each edge biconnected component, the maximum value and the minimum value can appear on the same cycle.
Assume that we have found the maximum edge and the minimum edge, then we can run a network flow to find two paths connecting the edges.
Deleting the two edges and connect them to S and T respectively will result in a correct network flow graph.
After finding the edges in the final answer, simply run Euler tour or carefully implemented DFS can solve the problem.
Note that if the range is 0, then some programs may find the same edge for the maximum and the minimum, which will result in WA.

Thank you!