Problem A. A Dirty Tricks

Note that $10^{10} + 9 = 33889 \cdot 295081$, so it it sufficient enough to precalc the answers for primes less than 295 081 (for greater primes the value of the product will be equal to zero). Note that `int128` or fast multiplication (implementation of the multiplication using modular addition) must be used in order to do the calculations given modulo.

Problem D. Dual Language Stable Triples

Note that the value of highest bit in c does not exceed the value of highest bit in a and b (because $c = a \oplus b$). But if $a \geq 2$ and $b \geq 2$, the highest bit of a^b exceeds the value of the highest bit in both a and b. If $a = 1$, then $a^b = 1$ for any b, but $1 \oplus b = 1$ means that $b = 0$. $b = 1$ gives no solution, because $a^1 = a$, but $a \oplus 1$ cannot be equal to a. $a > 1$ gives no solution ($a^0 = 1$ and $a \oplus 0 = 1$, so $a = 1$ only).

So only the triple (1, 0, 1) is stable.

Problem H. How to Guess Power of Two

We will precompute the sum of the digits of all powers of two, store the digit sums for each exponent, and create an associative array (map) for each achievable digit sum, where we will keep vectors of the original exponents.

We will make the first query with $d = 0$ to find out the sum of the digits of the number s_0. If the corresponding vector contains only one element, we immediately output the answer. Note that for digit sums equal to 1, 2, and 4, the answer is unique, meaning the need to ask a second question will only arise when $n \geq 3$, so subtraction is safe in terms of going out of bounds.

Otherwise, we check for k from 1 to 3 whether it is true that all numbers of the form $2^{v_i - k}$, where v_i are the exponents corresponding to s_0, have different digit sums. Upon finding the first such k, we make a second query with $d = -k$, after which the answer is determined unambiguously (moreover, the answers can be precomputed in advance at the stage of finding k).

It can be shown (for example, by checking all exponents in the range) that k from -3 to -1 will be sufficient to distinguish any two exponents within the problem's constraints.

Problem J. Just an ACM-subsequence

The problem is solved using dynamic programming.

To count the number of strings with ACM, we will construct a two-dimensional array `dp`, where the first parameter is the value n, and the second is the longest subsequence from the set $\{empty, \mathtt{A}, \mathtt{AC}, \mathtt{ACM}\}$ that exists in the string of length n, and the content of $dp_{i,j}$ is the number of strings of length i in which the j-th subsequence from the set can be found, but not larger (0: empty string, 1: contains `A`, but does not contain `AC`, and so on). The transition table is: $dp_{0,i+1} = dp_{0,i} \cdot 25$ (adding all letters except `A` is acceptable), $dp_{1,i+1} = dp_{1,i} \cdot 25 + dp_{0,i}$ (adding all letters except `C` is acceptable, plus adding `A` to the string from state 0), similarly $dp_{2,i+1} = dp_{2,i} \cdot 25 + dp_{1,i}$ and $dp_{3,i+1} = dp_{3,i} \cdot 26 + dp_{2,i}$. The answer will be in $dp_{n,3}$.

The total number of strings is 26^n, then we apply standard division by a prime modulus.

Problem K. Kingdom And Airlines

The airplane follows a estimated direction (in the definition of the problem) if and only if it is moving along a segment of the polyline between (x_i, y_i) and (x_{i+1}, y_{i+1}), which has the following property: the points (x_i, y_i), (x_{i+1}, y_{i+1}), and (x_n, y_n) lie on the same straight line. Therefore, read all the points and check that the oriented area of the parallelogram, that is, the cross product of the vectors $(x_i - x_n, y_i - y_n)$ and $(x_{i+1} - x_n, y_{i+1} - y_n)$ is equal to zero. If this is the case, add the length of the segment to the answer. Note that the answer will always be non-zero since for the segment before landing, one of the endpoints of the segment coincides with the point (x_n, y_n), and the length of this segment will definitely be counted.