Kick Start 2022 - Coding Practice with Kick Start Session #3

Analysis: Sherlock and Watson Gym Secrets

View problem and solution walkthrough video

The problem translates to given $\mathbf{A}$, $\mathbf{B}$, $\mathbf{N}$, $\mathbf{K}$, find number of pairs (i, j) which satisfy the following conditions:

1. $i \neq j$
2. $1 \leq i \leq \mathbf{N}$ and $1 \leq j \leq \mathbf{N}$
3. $(i^{\mathbf{A}} + j^{\mathbf{B}}) \ \ mod \ \ \mathbf{K} = 0$

Test Set 1

We can brute force through all possible pairs (i, j) which satisfy the first two conditions and calculate if $(i^{\mathbf{A}} + j^{\mathbf{B}}) \ \ mod \ \ \mathbf{K} = 0$. As $1 \leq i \leq \mathbf{N}$ and $1 \leq j \leq \mathbf{N}$, we have $\mathbf{N}^2$ pairs and for each pair we need to compute $(i^{\mathbf{A}} + j^{\mathbf{B}}) \ \ mod \ \ \mathbf{K}$, If we compute this naively, we will require $O(\mathbf{A} + \mathbf{B})$ operations, but we can use exponentiation by squaring to compute this efficiently in $O(log(\mathbf{A}) + log(\mathbf{B}))$.

Since we need to compute this sum for each pair (i, j), we get $O(\mathbf{N}^2(log(\mathbf{A}) + log(\mathbf{B})))$.

Test Set 2

As $\mathbf{N}$ is quite large for this test set, the previous approach would not work here, Let us try another approach. For $(i^{\mathbf{A}} + j^{\mathbf{B}}) \ \ mod \ \ \mathbf{K} = 0$, we know that $j^{\mathbf{B}} \ \ mod \ \ \mathbf{K} = -i^{\mathbf{A}} \ \ mod \ \ \mathbf{K}$.

For each possible i, let us try to find how many such j exists.
Let us create an array L, where $L[x]$ denotes the number of possible values of j such that $j^{\mathbf{B}} \ \ mod \ \ \mathbf{K} = x$. Now, we can iterate over all possible values of i i.e. $(1, 2, 3, \ldots, \mathbf{N})$ and add $L[-i^{\mathbf{A}} \ \ mod \ \ \mathbf{K}]$ to the answer. One more thing which is left to handle here is the condition $i \neq j$, for which we can simply check if $(i^{\mathbf{A}} + i^{\mathbf{B}}) \ \ mod \ \ \mathbf{K} = 0$ and decrement 1 from the answer.
The complexity of this approach would be $O(\mathbf{N}(log(\mathbf{A}) + log(\mathbf{B})))$ which is better than the previous approach but still exceeds the time limit. But we can try to optimize more from here.

You can note that even though i can take $\mathbf{N}$ possible values, $i \ \ mod \ \ \mathbf{K}$ can only take $\mathbf{K}$ possible values. Can we take advantage of this ?
We defined $L[x]$ = number of j such that $j^{\mathbf{B}} \ \ mod \ \ \mathbf{K} = x$. A more optimized way to construct L can be: Consider a variable $q, 0 \leq q < K$. (The number of values $j \ \ mod \ \ \mathbf{K}$ can take), we can iterate over all possible values of q i.e. $(1, 2, 3, \ldots, \mathbf{K})$ and increment $L[q^{\mathbf{B}} \ \ mod \ \ \mathbf{K}]$ by number of j's such that $j \ \ mod \ \ \mathbf{K} = q$. This reduces the time complexity of constructing L from $O(\mathbf{N}log(\mathbf{B}))$ to $O(\mathbf{K}log(\mathbf{B}))$.

Similarly, instead of iterating over all possible values of i, Consider a variable $p, 0 \leq p < \mathbf{K}$ (The number of values $i \ \ mod \ \ \mathbf{K}$ can take), Let z = number of i's such that $i \ \ mod \ \ \mathbf{K} = p$, and add $z \times L[-p^{\mathbf{A}} \ \ mod \ \ \mathbf{K}]$ to the answer.

We still need to handle the condition of $i \neq j$, for which we can simlpy check for each p if $(p^{\mathbf{A}} + p^{\mathbf{B}}) \ \ mod \ \ \mathbf{K} = 0$ and subtract z from the answer.

As we are iterating over p and q which take only $\mathbf{K}$ distinct values instead of $\mathbf{N}$ distinct values for i and j, the complexity of this approach reduces to $O(\mathbf{K}(log(\mathbf{A}) + log(\mathbf{B})))$.

A small mistake which can happen in exponention by squaring is $x^0 = 1$, which would be wrong in case of $\mathbf{K} = 1$, hence take $x^0 = 1 \ \ mod \ \ \mathbf{K}$.