5 Rarest Insects

Written by: **Hazem Issa** (Egypt), Egyptian Olympiad in Informatics (EOI)
Prepared by: Abdul Malik Nurrokhman
Solutions, review, and other problem preparations by: Hocky Yudhiono, Jonathan Irvin Gunawan, Prabowo Djonatan
Analysis author: Abdul Malik Nurrokhman

For this discussion, let D be the number of different types of insects, and ans be the cardinality of the rarest insects.

5.1 Subtask 1

In this subtask, $N \leq 200$.

Let x be an insect whose type is unknown. For all other insects y, we can check whether they have the same type by putting them (and only them) inside the machine and then pressing the button. If the button replied with 2, that means they are both the same type.

Do it for every pair of insects x and y to obtain the answer.

Query complexity: $O(N^2)$

We can do further optimization by ignoring some insects whose types are already known.

Query complexity: $O(N \times D)$

5.2 Subtask 2

In this subtask, $N \leq 1000$.

Let I be the number of insects inside the machine.

To know the value of D (the number of different types of insects), we need to put all insects one by one and press the button each time. If the machine reports one more than the previous press, then move the most recently inserted insect outside. When this is done, $D = I$.

Repeat the above process again for insects which are outside the machine. This time, our goal is to check whether the rarest cardinality is 2. If $I < 2D$ after this second process, then the answer is 1. Otherwise, repeat the process again to check whether the answer is $2, 3, \ldots$.

Query complexity: $O(N \times ans)$

Note that $ans \times D \leq N$, so $\min(ans, D) \leq \sqrt{N}$. If we combine the last two solutions above, then we can solve this subtask.

Query complexity: $O(N\sqrt{N})$

5.3 Subtask 3

In this subtask, $N \leq 2000$ and a partial score might be given.

5.3.1 $O(N \log N)$ query complexity

Let B be the upper bound of the answer. That means B is initially N.

To check whether b can be the bound, we insert the insects one by one, and move the recently inserted insect outside again whenever the press button replied with $> b$. If $I = b \times D$ after this process, then $ans \geq b$. We will call this check the *upper bound check*.

We can binary search the upper bound so we can get around 50 points.

5.3.2 $O(N \times \textbf{invfact}(N))$ query complexity

Let invfact(N) be the inverse factorial function.

We can initially set $B = \frac{N}{D}$. If $I = B \times D$ after each upper bound check, then the answer is B, else set $N = I$ and repeat the process. With this trick, we only need to check invfact(N) upper bounds. We will get around 70 points with this solution idea.

5.3.3 $O(3N)$ query complexity

From the binary search solution, we don't exactly need to re-empty the machine. After each iteration, if $I = B \times D$ then keep the insects inside the machine without removing them. Otherwise, if $I < B \times D$ then just remove all insects that are inserted in the last iteration. Therefore, the maximum operations needed for each type are divided by two for each iteration. Note that, $N + \frac{N}{2} + \frac{N}{4} + \cdots = 2N$. We also need an additional N operations in the beginning to get the value of D.

5.3.4 Constant optimization

The previous solution actually yields slightly more than $3N$ so we need further optimization. One of the optimizations that we can do is to stop moving insects inside the machine when $I = B \times D$.