Task

Chaos and Trouble are playing a game. Chaos begins by picking a positive integer that does not exceed N. Trouble then tries to find this integer: on each move, he asks Chaos a question of the form "Is your number one of a_1, a_2, ..., a_k?" and Chaos answers either *Yes* or *No*.

There is a catch, though: Chaos may decide at some point that the game is not interesting enough as it stands and start lying on every question to make things somewhat more exciting. So, Trouble knows that Chaos is going to answer all questions correctly up to a point and give only false answers after that, but he does not know how many true answers to expect. (If Chaos finds the game particularly entertaining, she may answer all questions correctly. On the other hand, if she decidedly *doesn't* find the game particularly entertaining, all of her answers could be false.)

Your task is to write a computer program that helps Trouble find Chaos's number by asking as few questions as possible.

Analysis and solution

For starters, here is what appears to be the most straightforward strategy for Trouble.

Let x be Chaos's number and $k = \lceil \log_2 N \rceil$. For his first k questions, Trouble implements a simple binary search: his i-th question should be equivalent to "What is the i-th binary digit of $x - 1$?".

Following this, on move $k + 1$ Trouble asks some question that he knows the answer to, e.g., "Is x greater than zero?". If Chaos answers correctly, then all of her previous answers must have been correct as well and Trouble is done. If not, then Trouble knows that all of Chaos's subsequent answers are going to be false – which is just as well as them being true, as he can simply flip the answers in his head.

Furthermore, Trouble knows that Chaos has answered the first i questions honestly and that she has lied on the next $k-i$ questions for some $0 \leq i \leq k$. Each option for i uniquely determines the true answers to the binary search questions, leading to at most one possibility x_i for x. A second binary search over just x_0, x_1, ..., x_k, and Chaos's number is found.

So, $k + 1 + \lceil \log_2(k + 1) \rceil$ questions suffice. Can Trouble do any better? As it turns out, yes, but in order to see how we need to analyse the game somewhat more deeply.

Consider the situation that Trouble finds himself in right after Chaos's first answer. He knows that either (a) she answered correctly, her number is in some subset A of $\{1, 2, \ldots, N\}$, and her subsequent answers are going to follow the same rules as if the game had just started, or (b) she lied, her number is in the complement B of A, and all of her subsequent answers are going to be false.

The key observation is that this situation self-replicates!

More precisely, let $G(A, B)$ be the situation in which Trouble knows that either (a) Chaos's number is in the set A and her subsequent answers are going to follow the same rules as if the game had just started, or (b) Chaos's number is in the set B and her subsequent answers are all going to be false. (In all situations relevant to our analysis, the sets A and B are going to be disjoint.)

Suppose that Trouble next asks Chaos if her number is in the set C. Write C as $C = A_1 \cup B_1$ where $A = A_1 \cup A_2$ is a partitioning of A and $B = B_1 \cup B_2$ is a partitioning of B.

Suppose that Chaos answers *Yes*. Then either she is telling the truth and her number is in A_1, or she is lying and her number is in $A_2 \cup B_2$. This means that we have entered the situation $G(A_1, A_2 \cup B_2)$.

Suppose, on the other hand, that Chaos answers *No*: analogous reasoning shows that in this case we find ourselves in the situation $G(A_2, A_1 \cup B_1)$.

What Trouble needs to do is reach a situation $G(A, B)$ such that either $|A| = 1$ and $|B| = 0$ or $|A| = 0$ and $|B| = 1$, and in as few steps as possible. How does he do so?

In order to make things somewhat easier for Chaos (and ourselves), let us augment the rules of the game a bit: Chaos does not need to pick a specific number in advance anymore, she just needs to give answers that are consistent with the rules of the game and with at least one *possibility* for a number that she could have picked. Clearly, the worst-case scenario for Trouble is precisely the same in both games. Furthermore, when analysing a situation $G(A, B)$, we can obviously do away with the specific sets A and B and retain only the number of elements a and b in each.

With that said, the game starts to look as follows: in the beginning, Trouble and Chaos are in the situation $G(N, 0)$. On each turn, given a situation $G(a, b)$, Trouble chooses two partitionings $a = a_1 + a_2$ and $b = b_1 + b_2$, offers Chaos a choice between the situations $G(a_1, a_2 + b_2)$ and $G(a_2, a_1 + b_1)$, and Chaos picks one (though not $G(0, 0)$, if it is an option at all). Trouble strives to reach $G(1, 0)$ or $G(0, 1)$ as quickly as possible, and Chaos is doing her best to prevent this.

Let $h(a, b)$ be the number of moves that Trouble needs given best play by Chaos. The preceding paragraph gives us a recurrence relation for $h(a, b)$:

$$h(a,b) = 1 + \min_{\substack{a=a_1+a_2 \\ b=b_1+b_2}} \max\{h(a_1, a_2 + b_2), h(a_2, a_1 + b_1)\}.$$

This can be rewritten as

$$h(a,b) = 1 + \min_{\text{appropriate } (p,q) \text{ and } (u,v)} \max\{h(p,q), h(u,v)\},$$

where "appropriate" means that (p, q) and (u, v) are a pair of symmetric points within the parallelogram $P(a, b)$ of vertices $(0, a)$, $(a, 0)$, $(0, a + b)$, and (a, b).

This recurrence suffices to compute $h(a, b)$ for all a and b. Indeed, order all (a, b) in ascending order by $a+b$ and, in the case of a tie, in descending order by b: then evaluating $h(a, b)$ is always reduced to evaluating $h(x, y)$ at points (x, y) that precede (a, b). Let us compile a table for $h(a, b)$ [See Table 1].

b \ a	0	1	2	3	4	5	6	7	8	9
0	↯	0	3	5	5	6	6	6	6	7
1	0	2	4	5	5	6	6	6	6	7
2	1	3	4	5	5	6	6	6	6	7
3	2	3	4	5	5	6	6	6	6	7
4	2	3	4	5	5	6	6	6	6	7
5	3	4	4	5	5	6	6	6	6	7
6	3	4	4	5	5	6	6	6	6	7
7	3	4	5	5	5	6	6	6	6	7
8	3	4	5	5	5	6	6	6	6	7
9	4	4	5	5	6	6	6	6	7	7

Table 1

Notice that we do not yet know how to compute $h(a, b)$ very efficiently: the recurrence relation requires that we comb through about $\frac{1}{2}ab$ possibilities until we know where the minimum is attained, so if we desire to find $h(a, b)$ then generating the relevant portion of the table is going to take a number of steps on the order of $(a + b)^4$.

A natural guess would be that the minimum is always attained at the center of $P(a, b)$. Translating this back to a strategy for Trouble, he should always play so that a_1 and a_2 are as

close as possible to $\frac{a}{2}$ and b_1 and b_2 are as close as possible to $\frac{b}{2}$. (In the case when a and b are both odd, this does not determine Trouble's move uniquely. Experimenting with small values for a and b suggests that we should take $a_1 = \lfloor \frac{a}{2} \rfloor$, $a_2 = \lceil \frac{a}{2} \rceil$, $b_1 = \lfloor \frac{b}{2} \rfloor$, and $b_2 = \lceil \frac{b}{2} \rceil$.) Actually, this guess is incorrect and the strategy that it leads to is not optimal – but it is still better than the naïve strategy that we discussed in the beginning!

Let us investigate further. The values in each column appear to grow very slowly, so it makes sense to compress the table by retaining only the points where those values increase strictly. To this end, let $c(a,k)$ be the largest b such that $h(a,b) \leq k$. The recurrence relation for $h(a,b)$ leads to the following recurrence relation for $c(a,k)$:

$$c(a,k) = \max_{\substack{u+v=a \\ u \leq c(v,k-1) \\ v \leq c(u,k-1)}} [c(u,k-1) + c(v,k-1)] \; - \; a,$$

and the table for $c(a,k)$ looks like this [See Table 2]. (Here, $c(a,k) = \times$ means that $h(a,b) > k$ for all b.)

k \ a	0	1	2	3	4	5	6	7	8	9
0	1	0	×	×	×	×	×	×	×	×
1	2	0	×	×	×	×	×	×	×	×
2	4	1	×	×	×	×	×	×	×	×
3	8	4	0	×	×	×	×	×	×	×
4	16	11	6	×	×	×	×	×	×	×
5	32	26	20	14	8	×	×	×	×	×
6	64	57	50	43	36	29	22	15	8	×
7	128	120	112	104	96	88	80	72	64	56
8	256	247	238	229	220	211	202	193	184	175
9	512	502	492	482	472	462	452	442	432	422

Table 2

The pattern is now clear: the numbers in row k form an arithmetic progression that starts at 2^k and decreases by $k+1$. Given the recurrence relation for $c(a,k)$, this is easily verified by induction. When, though, does the progression stop?

Consider the last non-$\times$ term $c(m_k,k)$ in row k. The recurrence relation tells us that there are $u \leq v$ such that $m_k = u+v$, $c(u,k-1) \geq v$, and $c(v,k-1) \geq u$. Some experimentation with small numbers suggests that we should turn our attention to the last term $c(s_{k-1},k-1)$ in row $k-1$ such that $c(s_{k-1},k-1) \geq s_{k-1}$: setting $u = v = s_{k-1}$ gives us $m_k \geq 2s_{k-1}$, and the bound is tight for $3 \leq k \leq 9$. Is there any $k \geq 10$ such that $m_k > 2s_{k-1}$? Let us check.

If $v \leq s_{k-1}$, then $m_k = u + v \leq 2s_{k-1}$. If $v > s_{k-1}$, then $m_k = u + v \leq c(v,k-1) + v \leq c(s_{k-1}+1,k-1) + s_{k-1} + 1$ (since $c(w,k-1)+w$ is a strictly decreasing function of w) $\leq 2s_{k-1}+1$ (since $s_{k-1}+1 > s_{k-1}$ gives us $c(s_{k-1}+1,k-1) < s_{k-1}+1$ by the definition of s_{k-1}). Therefore, the only way that we can have $m_k > 2s_{k-1}$ is if $c(s_{k-1}+1,k-1) = s_{k-1}$, $u = s_{k-1}$, $v = s_{k-1}+1$, and $m_k = 2s_{k-1}+1$.

Since

$$c(a,k) = 2^k - a(k+1)$$

as long as $a \leq m_k$, this allows us to put together and prove by induction precise formulas for s_k and m_k: for all k

$$s_k = \left\lfloor \frac{2^k}{k+2} \right\rfloor$$

and for all $k \geq 1$

$$m_k = 2 \left\lfloor \frac{2^{k-1}}{k+1} \right\rfloor + e_k$$

where $e_k = 0$ if $k+1$ does not divide $2^{k-1}+1$ and $e_k = 1$ otherwise. (It is not immediately obvious if there are any $k \geq 3$ such that $k+1$ divides $2^{k-1}+1$. As it turns out, there are, and the least one is $k = 49736$.) The explicit formula for $h(a,b)$ is then

$$h(a,b) \text{ is the least positive integer } k \text{ such that } a \leq m_k \text{ and } a(k+1)+b \leq 2^k.$$

What does all of this tell Trouble (who by this point has potentially become quite confused) about optimal play? Once he has calculated $k = h(a,b)$, he needs to find a partitioning $a = a_1 + a_2$ such that $c(a_1, k-1) \geq a_2$ and $c(a_2, k-1) \geq a_1$. For any such partitioning, he would then be able to find $b = b_1 + b_2$ such that $c(a_1, k-1) \geq a_2 + b_1$ and $c(a_2, k-1) \geq a_1 + b_2$. Since taking a_1 and a_2 to be as close to $\frac{a}{2}$ as possible always works, he might as well do that: so, set $a_1 = \left\lfloor \frac{a}{2} \right\rfloor$ and $a_2 = \left\lceil \frac{a}{2} \right\rceil$. As to the partitioning of b, for a even clearly $b_1 = \left\lfloor \frac{b}{2} \right\rfloor$ and $b_2 = \left\lceil \frac{b}{2} \right\rceil$ does the job, whereas for a odd it is easy to verify that we can safely set $b_1 = \left\lfloor \frac{b-k+1}{2} \right\rfloor$ and $b_2 = \left\lceil \frac{b+k-1}{2} \right\rceil$.

As a finishing touch, let us look at the actual optimal number of questions that Trouble needs to find Chaos's number. Substituting $a = N$ and $b = 0$ in the formula for $h(a,b)$, we see that for $N \geq 2$ this number is the least positive integer k such that

$$N \leq 2 \left\lfloor \frac{2^{k-1}}{k+1} \right\rfloor + e_k$$

where e_k is defined as above. Asymptotically,

$$k = \log_2 N + \log_2 \log_2 N + O(1),$$

same as in both the naïve and improved strategies!

[How do the naïve and improved strategies compare to the optimal one, in more precise terms?]

[What about the variant of the game in which Trouble has to precommit to the series of questions that he is going to ask?]

[What about the variant of the game in which Chaos switches between telling the truth and lying a bounded number of times? What is the asymptotics of the optimal number of questions in this game?]

[What about the variant of the game in which Chaos may lie on no more than p questions out of every q successive ones, e.g., with $p = 1$ and $q = 3$?]