A hint for the extra credit question

Recall the problem: Alice and Bob play the following number search game. \( N \) is fixed to be some large number. Alice picks an integer \( y \) in the range \([1 \cdots N]\). Bob’s goal is to guess this number as quickly as possible through a series of questions. At step \( t \) Bob poses a question \( x_t \), also an integer in the range \([1 \cdots N]\). For \( t \geq 2 \), Alice responds by “hotter” or “colder” depending on whether \( x_t \) is closer or farther from her number \( y \) compared to the previous question \( x_{t-1} \). That is, if \( |x_t - y| < |x_{t-1} - y| \), Alice responds “hotter”; if \( |x_t - y| > |x_{t-1} - y| \), Alice responds “colder”; and if the two are equally far, Alice responds “equal”. The game ends as soon as Bob correctly guesses \( y \).

Our goal is to give a strategy for Bob that finds \( y \) in at most \( \log_2 N + O(1) \) steps.

Ground work First we’ll talk about Alice. Suppose that Alice’s goal is to make Bob take as long as possible. Then, instead of thinking about a number up-front, she makes up the answers on the go, so that at least one number satisfies those answers and the game runs for as long as possible.

Let \( S \) be the set of numbers that Bob has not eliminated yet. At every step, when Bob asks a question, \( S \) gets divided into three parts, \( S_h \), \( S_c \), and \( S_e \), the numbers for which Alice’s answer would be “hotter”, those for which her answer would be “colder”, and those (exactly one) for which her answer would be “equal”. Alice can pick her answer based on which of these three sets is the largest. So we can assume that at every step the size of \( S \) can at best get halved, and be no smaller.

Note that \(|S|\), i.e. the number of numbers that haven’t been eliminated yet, starts out as being \( N \) and ends up being 1. Since we are aiming for at most \( \log_2 N + O(1) \) rounds, this means that we should aim for \(|S|\) to get halved at almost every step.

The “unconstrained strategy” In the game, Bob is not allowed to ask as a question any number outside of the range \([1 \cdots N]\). Suppose for the moment that he was allowed to ask any number as a question (e.g. arbitrarily large, or arbitrarily small negative numbers). Can you then design a strategy that guesses the answer in \( \log_2 N + O(1) \) steps?

The overall strategy Our overall strategy will have two parts. At the end of the first part, we will aim to make \( S \) small enough and far enough from 1 and \( N \), so that in the second part we can run the unconstrained strategy without worrying about having to use numbers outside of \([1 \cdots N]\). How far from 1 and \( N \) does \( S \) have to be to enable this? In the first part, we want to reduce the size of \( S \) by a factor of 2 at every step, until it satisfies the properties we need for the second part. How can we do this?