COVID Editorial

Subtask 1

Almost every correct solution should pass. For example we can test the students one by one.

Binary search

Approximately 35 points could be achieved using binary search: Start with all students as one segment. If the whole segment is positive, test left half and right half separately.

Slightly more points can be achieved using the following observation: When the entire segment is positive and its left half is negative, the right half must be necessarily positive.

Divide to blocks

For each P, we determine a block size. Next, we divide the input into blocks and in each of them we do binary search. Suitable size of a block is one where the probability of a positive block is around 0.5. We can get about 40 points with this approach.

Instead of finding all positive students in a block, we can find the first positive. That guarantees one query per level. After finding them, we start new block immediately after this student. This optimization yields 95 points. Remaining 5 can be achieved by fine-tuning this solution.

Optimal solution using dynamic programming

We will consider set of all solutions of following format: All unprocessed students will be a suffix of the input. First, we ask for a prefix of this suffix. If there is no one positive, we shorten the suffix. Otherwise, we continue queries until we find the first positive in the prefix. When we find the first positive, we shorten the suffix.

For finding the first positive, we always query a prefix of a segment in which we know that there is someone positive. If there is someone positive in the prefix, we shorten the queried prefix. If there is no one, we also shorten the unprocessed suffix. When the segment has length one, we know that we have found the first positive.

We can notice that improved binary search is a solution of this type.

From this set, we can use dynamic programming to compute the solution with the lowest $\mathbb{E}[Q]$ — expected value of queries. We can compute this from table of $\mathbb{E}[Q]$ for all possible lengths of an unprocessed suffix and a prefix containing somebody positive.

We start from the shortest lengths of unprocessed suffix and in each step we first compute cases with a known positive prefix from the smallest to the largest. Then we compute the state where we only do not know the positive prefix. We always try all possible lengths of a query. From probability of a positive query and from the previous values we determine the expected value for this query.

If we accompany each cell of the table by which query yields the optimum, we can reconstruct the algorithm.