EJOI 2021, Day 2,
English Editorial

August 27, 2021

Problem 1: Binsearch

(Proposed by Maria-Alexa Tudose.)

Let us call the value $\frac{n-1}{2}$ "the middle value" and the position $\frac{n-1}{2}$ (in the permutation p) "the middle position".

Subtask 1. When $b_i = true$ for all i, we can achieve $S(p) = 0$ by setting p to be the increasing permutation.

Subtask 2. When $b_i = false$ for all i, we can achieve $S(p) = 1$ in the following way:

- Place the middle value on the middle position.
- Place all values smaller than the middle value on positions $\frac{n-1}{2} + 1, \ldots, n$, in any order.
- Place all values bigger than the middle value on positions $1, \ldots, \frac{n-1}{2} - 1$, in any order.

In particular, note that the decreasing permutation works.

Subtask 3. In this subtask $N = 7$ holds. We can use the fact that $7! = 5040$ is small. This allows us to generate all the possible permutations of size n, and for each such permutation we can run binary search on all values from 1 to n and compare the returned result with the desired value in b. We calculate $S(p)$ for all permutations p, and print any permutation p that achieves $S(p) \leq 1$.

Subtask 4. This subtask encourages solutions with sub-optimal (but polynomial) time complexities. For example, a poor implementation of the "Solution 1" presented below might have a time complexity of $O(n^2)$ instead of the optimal $O(n)$.

Subtask 5. In this subtask, the sequence b is guaranteed to be generated randomly. This allows us to design solutions which use different facts, such as:

- The numbers of ones and zeros in b should be approximately equal.
- There are not many consecutive equal entries in b.

Subtask 6. The problem admits a wide variety of full solutions which promote different types of thinking. We present only a few of them.

Solution 1. We try to place, in turn, each possible value on the middle position. Let X be our current try. Our aim is to find p for which $b_i =$ `binary_search(n, p, i)` for all values $i \neq X$. This would give $S(p) = 0$ if $b_X = 1$ and would give $S(p) = 1$ if $b_X = 0$.

We define two sets L and R:

- $L = \{i \mid b_i = \texttt{true}, i < X\} \cup \{i \mid b_i = \texttt{false}, i > X\}$
- $R = \{i \mid b_i = \texttt{true}, i > X\} \cup \{i \mid b_i = \texttt{false}, i < X\}$

We also define two sets A and B:

- $A = \{i \mid b_i = \texttt{true}\}$
- $B = \{i \mid b_i = \texttt{false}\}$

Lemma. *If $|L| = |R| = \frac{n-1}{2}$ for some X, then we can find a good permutation.*

Proof. One way to build such a permutation is:

- First place the values in L in increasing order
- Then place the values in R in increasing order

□

Lemma. *There exists an X such that $|L| = |R| = \frac{n-1}{2}$.*

Proof. We aim to achieve $|L| = \frac{n-1}{2}$.

When we change X to $X + 1$, $|L|$ either stays constant, increases by 1, or decreases by 1.

For $X = 1$, we have $|L| = |B - \{1\}|$. Also, when we set $X = n$, we have $|L| = |A - \{n\}|$. Therefore, $|L| \leq \frac{n-1}{2}$ for $X = 1$ or for $X = n$, and $|L| \geq \frac{n-1}{2}$ for $X = 1$ or for $X = n$. Combining these observations, we get the conclusion. □

Solution 2. If $A = \emptyset$, then we return the decreasing permutation.

From now on, assume that $A \neq \emptyset$

We choose X to be any value from A for which $|L \cap A|, |R \cap A| \leq \frac{n-1}{2}$.

Intuitively, this means that we place on the middle position any value from A for which all remaining values from A “fit properly” in the remaining halves.

There are 2 cases now: $|L| \geq |R|$ and $|L| < |R|$. We will treat only the first case, because the second one can be solved symmetrically.

Assuming $|L| \geq |R|$, let W be a set of size $|L| - \frac{n-1}{2}$ s.t. $W \subseteq L \cap B$. We will place the values in $L - W$ in the first half of the permutation in increasing order and the values in $R \cup W$ in the second half (in an order to be determined).

All elements in the first half will get the required result when calling binary search. The value on the middle position also gets the required result.

We therefore need to arrange the elements in the second half such that we get at most one position i for which $b_i \neq$ `binary_search(n, p, i)`. We can solve this recursively.

Solution 3. (This solution is due to Tamio-Vesa Nakajima.) We will create a procedure $solve(A, B)$ which creates a sequence p for which $S(p) \leq 1$ if A is the set of indices i with $b_i = \texttt{true}$ and B is the set of indices i with $b_i = \texttt{false}$. This will be a recursive procedure, using the following cases:

- If $A = \emptyset$ then we are in subtask 2 – output B sorted in decreasing order.
- If $|A| + |B| \leq 3$ then the answer can be easily found by hand (there are only 8 cases).
- Now suppose $A \neq \emptyset$ and $|A| + |B| \geq 7$. Thus at least one element should be “found”. We have two further cases:

- Suppose $|A| > |B|$. Suppose there are $2^k - 1$ elements overall. Let A' contain the first 2^{k-1} elements of A in increasing order. Then output A' in increasing order first, followed by $solve(A - A', B)$. For example if $A = \{1, 3, 5, 6, 7\}, B = \{2, 4\}$, then $A' = \{1, 3, 5, 6\}$ and our output starts with $1, 3, 5, 6$ followed by the result of $solve(A - A' = \{7\}, B = \{2, 4\})$.
- Suppose $|A| < |B|$. Suppose $|A| + |B| = 2^k - 1$. (Note that the case $|A| = |B|$ is impossible as $|A| + |B| = 2^k - 1$ is odd.) Since $|B| > |A|$ we deduce that $|B| \geq 2^{k-1} - 1$. Let $t = 2^{k-2} - 1$. Let X be the first t elements of B in increasing order and Y be the last t elements of B in increasing order. Since $|B| \geq 2t$ we deduce that $X \cap Y = \emptyset$ i.e. X and Y have no common elements. Let b be an arbitrary element from $B - X - Y$. There are two cases: either $b < \min A$ or $b > \min A$ – we will assume without loss of generality that $b < \min A$, since the other case is treated symmetrically. We construct our array as follows:
  * The first t elements of the result are Y (in any order).
  * The next element of the result is b.
  * The next t elements of the result are X (in any order).
  * The next element (in fact the middle element) should be $a = \min A$.
  * The second half of the result should be $solve(A - \{a\}, B - X - Y - \{b\})$.

  Observe that:
  * All of the elements in Y are greater than b, and thus are not found.
  * $b < \min A$ and thus is not found.
  * All of the elements in X are less than b and thus are not found.
  * $\min A$ is immediately found.
  * By the correctness of *solve* the elements in the second half contribute at most 1 to $S(p)$.

  As an example, suppose $A = \{3, 4\}, B = \{1, 2, 5, 6, 7\}$. Then $X = \{1\}, Y = \{7\}, b = 2$, and $2 = b < \min A = 3$. Thus the array begins with $7, 2, 1, 3$ followed by the result of $(\{4\}, \{5, 6\})$, which can be $5, 4, 6$ for instance. Thus the result is $7, 2, 1, 3, 5, 4, 6$, with $S(p) = 1$.