CodeForces Round 1037 Div. 2C / Div. 1A

omeganot

July 20, 2025

We can represent the mode of transportation each day as a binary string, where a 0 corresponds to the bus and a 1 corresponds to the telescopic jet bridge.

If there is any element that appears at least 4 times in a, then the answer is YES because Vadim can cover every possible scenario over a two-day period (there are exactly four scenarios: 00, 01, 10, 11).

If this is not the case, then the frequencies of each day are less than 4. We may note that if a certain day a_i appears twice, then Vadim can force day $a_i + 2$ to be either 0 or 1. For example, he can force it to be 0 by choosing to bet 11 and 01 with his two students on day a_i. Then, if Vadim has at least one bet for day $a_i + 1$, Vadim can continue this trend by forcing day $a_i + 3$ to be 0 by selecting his $a_i + 1$ bet to be 01. This may continue until Vadim finally reaches another day where he has two bets, and then he can cover both 00 and 01. Note that, if at any day in between, Vadim has no more bets, then this chain is interrupted as he will be unable to force the next couple of days to be either 0 or 1.

It seems like there is no answer if Vadim cannot employ the above strategy, so let's try to prove that. Consider the infinite array b where b_i equals the frequency of i in the array a. In this infinite array b, if none of the subarrays without only nonzero elements can guarantee a victory for Vadim, then b cannot guarantee a victory for Vadim, since if $b_i = 0$ then Vadim can't possibly win on days $i + 1$ and $i + 2$, and he can't force day $i + 2$ to be 0 or 1 either, essentially making bets from day $i + 1$ onwards completely separate to anything that came before it.

Thus, if we consider the subarrays without zeros after removing all the zeros, we notice they take the form of either $1, 1, \ldots, 1$ or $1, \ldots 1, 2, 1 \ldots, 1$, or $1, \ldots 1, 3, 1, \ldots, 1$. Let's say the subarray (l, r) consists only of 1's. Then, for any sequence of bets Vadim makes, we can construct a string which prevents him from winning using the following strategy:

Consider building the string from days l to r starting at l, and we will be creating the string on days $l + 1$ to $r + 2$. If on day l, Vadim bets 00 or 01, then

we should should have 1 on day $l+1$. Thus, day $l+2$ isn't forced. If Vadim bets 10 or 11, then we should have 0 on day $l+1$ and achieve a similar result. Then, we look at the bet on day $l+1$ and employ a similar strategy when setting the value on day $l+2$. This continues until we process the bet on day r and the last element at position $r+2$ can be anything.

Now, consider the case where (l, r) is all 1's except $b_m = 2$ or $b_m = 3, (1 \leq m \leq r)$. We can still show that for any sequence of bets we can make Vadim lose. On days $l+1$ to m, we follow the previously detailed strategy on bets (l, m). Furthermore, for our string's values from $(m+3, r+2)$, we follow the previously detailed strategy on days $(m+1, r)$, except we will reverse the logic and start from day r instead. Since our aforementioned strategy means the last day we can bet whatever we want, there is no way for b_m ¡ 4 bets to successfully guarantee a win on days $m+1$ and $m+2$. My explanation is confusing but some scratch work may reveal the underlying ideas.

Thus, the previously mentioned condition is both necessary and sufficient. To check this condition, we can sort a and then use a two-pointers method to retrieve each subarray where the elements are consecutive (like $[1, 2, 2, 3], [5, 5, 5], [8, 9, 10]$). All we have to ensure is that there are no 4 adjacent elements that are the same and that for any of our derived subarrays, at most one element appears twice. Our Time Complexity is $O(n \log n)$ due to sorting.