Code Jam 2022 - Round 3

Analysis: Duck, Duck, Geese

Test Set 1

Given the lower limits for $\mathbf{N}$, we can check all possible contiguous subsets (as long as we check each one quickly). One way is to iterate through all possible starting indices for the contiguous subset. Then, for each starting index, we can loop through all possible sizes in order.

We can keep track of the counts of each color as well as the number of hat colors have counts that are invalid (not $\mathbf{0}$ and not in the acceptable range for that color). Note that when we extend our subset, only the hat color that the newly added child has is affected. If this changed whether or not this color is valid, we update the count of valid and invalid colors as needed. Then, if this count is equal to $\mathbf{C}$, we can increment our total answer (as long as the subset is of length at least 2 and at most $\mathbf{N} - 1$).

Because checking each contiguous subset is done in $O(1)$ time and there are $O(\mathbf{N}^2)$ subsets to check, our time complexity for Test Set 1 is $O(\mathbf{N}^2)$.

Test Set 2

For Test Set 2, $\mathbf{N}$ is too large to check each contiguous subset separately. We can speed this up by using a Segment Tree to check all possible subset lengths at once (for each starting index). We can remove the cyclic part of the problem by appending the array to itself.

Given a starting index, S, each color has two (possibly empty) ranges of ending indices that are valid for this color (meaning the number of hats of this color is either 0 or in the acceptable range). If we use add 1 to our segment tree for each value in these ranges and for all colors, we can count the number values (ending indices) in our segment tree for the range $[S+1, S+\mathbf{N}-2]$ that have a value of exactly $\mathbf{N}$. This count will tell us how many contiguous subsets are valid for our current start index.

Now, all we need to do is update our segment tree when moving the starting index to the right. Notice that moving our starting index to the right by one will only affect the valid end index ranges for the hat color of the child being removed (the one that previously was our starting index). If we precompute for each position, where is the next occurrence of that child's hat color, we can compute how the valid ranges for this color will move in $O(1)$. The exact implementation of this is left as an exercise to the reader.

Given that our segment tree operations each take $O(\log \mathbf{N})$ time, we can count the number of contiguous subsets for each starting index and move the starting index to the right both in $O(\log \mathbf{N})$ time. This gives us a final time complexity of $O(\mathbf{N} \log \mathbf{N})$.