Solution

At first, let's solve the problem when there is only one deque. A prefix value of deque will pop as the front of the deque and the rest of them will pop as back of deque. Let x of values pop as the front of the deque in the most powerful array we can make. The answer is at most the size of the longest increasing sequence (LIS) of the first x elements of the deque plus the size of the longest decreasing sequence of the last $n - x$ elements of the deque. We will show the answer is equal to this value too.

Let $\langle a_1, a_2, ..., a_k \rangle$ be our deque. Define dp_i as the longest increasing sequence of $\langle a_1, a_2, ..., a_i \rangle$, also define pd_i as the longest decreasing sequence of $\langle a_i, a_{i+1}, ..., a_k \rangle$. Let m be the maximum value of $dp_i + pd_{i+1}$ for each $1 \leq i < k$. We know that the answer is not greater than m. We have to make an array with power m. Let i be the one of the indexes which $dp_i + pd_{i+1} = m$, A be the elements of the longest increasing sequence of $\langle a_1, a_2, ..., a_i \rangle$ and B be the elements of the longest decreasing sequence of $\langle a_i, a_{i+1}, ..., a_k \rangle$. Concat the A and B together and name it C. You can easily make an array that the order of C come in the sorted way.

Now let's solve the general problem. Let ans_i be the most powerful array we can make using only i-th deque elements. It's obvious that the answer is not greater than $|ans_1| + |ans_2| + ... + |ans_n|$. We can make an array that has elements of $ans_1, ans_2, ..., ans_n$ in sorted order. To do that, at first we insert all of them into a set. Then until the set is not empty, we erase the minimum value of the set x from it. Let x be in the i-th deque. We will pop the front of i-th deque until we reach x in case that in ans_i, x has popped from the front. In the other case, we will do the same but pop the back of i-th deque.

The total time complexity is $O(nlg(n))$.