Queue Removals

You're given a list of `n` integers `arr`, which represent elements in a queue (in order from front to back). You're also given an integer `x`, and must perform `x` iterations of the following 3-step process:

Pop `x` elements from the front of queue (or, if it contains fewer than `x` elements, pop all of them)

Of the elements that were popped, find the one with the largest value (if there are multiple such elements, take the one which had been popped the earliest), and remove it

For each one of the remaining elements that were popped (in the order they had been popped), decrement its value by 1 if it's positive (otherwise, if its value is 0, then it's left unchanged), and then add it back to the queue

Compute a list of `x` integers `output`, the `i`th of which is the *1-based index in the original array* of the element which had been removed in step 2 during the `i`th iteration.

Signature

```
int[] findPositions(int[] arr, int x)
```

Input

`x` is in the range `[1, 316]`.
`n` is in the range `[x, x*x]`.
Each value `arr[i]` is in the range `[1, x]`.

Output

Return a list of `x` integers `output`, as described above.

Example

`n` = 6
`arr` = [1, 2, 2, 3, 4, 5]
`x` = 5
`output` = [5, 6, 4, 1, 2]

The initial queue is [1, 2, 2, 3, 4, 5] (from front to back).

In the first iteration, the first 5 elements are popped off the queue, leaving just [5]. Of the popped elements, the largest one is the 4, which was at index 5 in the original array. The remaining elements are then decremented and added back onto the queue, whose contents are then [5, 0, 1, 1, 2].

In the second iteration, all 5 elements are popped off the queue. The largest one is the 5, which was at index 6 in the original array. The remaining elements are then decremented (aside from the 0) and added back onto the queue, whose contents are then [0, 0, 0, 1].

In the third iteration, all 4 elements are popped off the queue. The largest one is the 1, which had the initial value of 3 at index 4 in the original array. The remaining elements are added back onto the queue, whose contents are then [0, 0, 0].

In the fourth iteration, all 3 elements are popped off the queue. Since they all have an equal value, we remove the one that was popped first, which had the initial value of 1 at index 1 in the original array. The remaining elements are added back onto the queue, whose contents are then [0, 0].

In the final iteration, both elements are popped off the queue. We remove the one that was popped first, which had the initial value of 2 at index 2 in the original array.