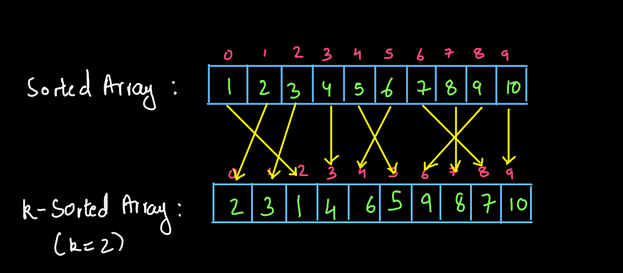
**1. Understanding Problem:**

This question says that we will be given an array and an integer k. This integer denotes that this array is k-sorted. We have to sort this k sorted array. So, what is a k-sorted array? Have a look at the diagram given below:

****

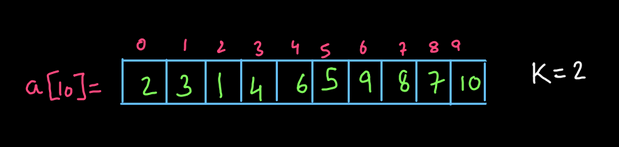
The diagram above shows the meaning of a k sorted array. In a k-sorted array, every element can move up to k positions away from its sorted position in both left and right directions. For example: in the diagram above, 1 has moved to k positions i.2. 2 positions to its right and 2 and 3 have moved 1 position i.e. k-1 position to their left. It may happen that the element does not move. For instance, in the above diagram, 4 and 10 have not changed their positions. So, we will be given such an array which will be k-sorted (k will be taken input and the array also) and we have to sort that array. You may refer to the question video if you have any doubts about the question. We recommend you try to solve this problem on your own first and then move to the solution.

Algorithm:

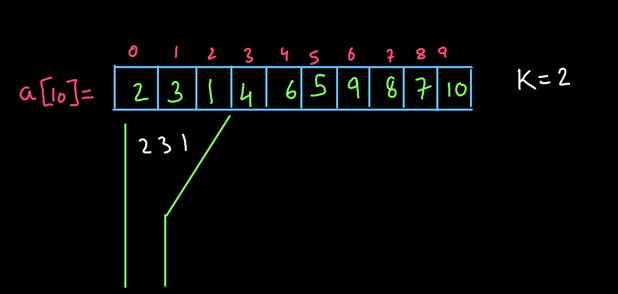
We create a priority queue of size k+1. We insert the first k+1 elements in it. Now, we remove one element from the priority queue. This will be the smallest element as the removal from the priority queue gives the smallest element or the element of highest priority. Now, we have k elements remaining in the priority queue. So, we insert the next element in the priority queue and the size again becomes k+1. Now we repeat the same procedure until we cover the entire array. If the array is complete and some elements are left in the priority queue, we empty the priority queue completely and stop the procedure.

**2. Approach:**

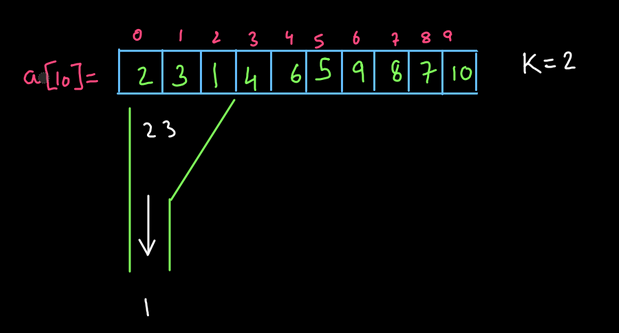
Let us say, we have an array given below. We are also given k=2 i.e. this array is 2-sorted.

****

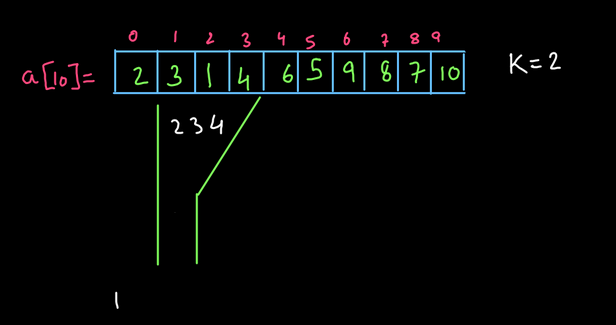
Now have a look at the diagram given below:

****

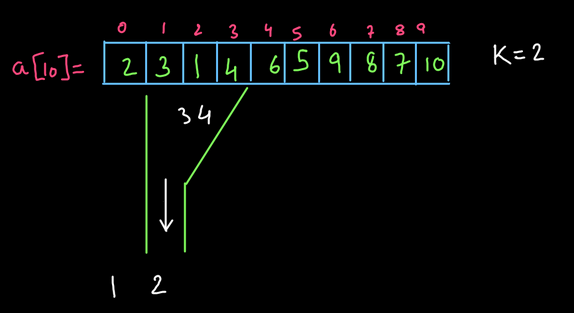
Let us say that we have a funnel. This funnel can filter out the element of least value. So, we made a funnel of size k+1=2+1=3 in this case and kept it at the 0th index. So, it can accommodate three elements 2,3 and 1. Since this funnel can filter out the smallest element, we get element 1 out from this funnel. (see fig-4)

****

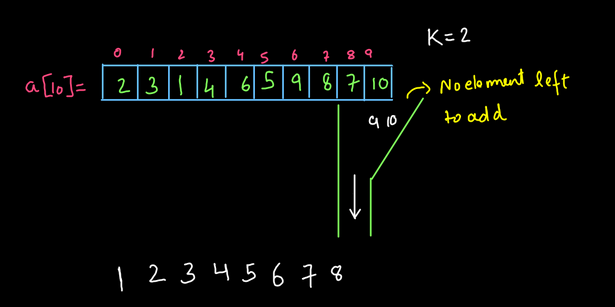
After this, the funnel has only two elements inside it. So, we can put one more element into this funnel. Let's move this funnel one step ahead and put one more element into it.

****

Now again, we will use this funnel's property to separate out the smallest element and we get another element in its sorted order.

****

So, we can apply the same technique till we reach the end of the array. We have to keep repeating this procedure until we have nothing left to put into this funnel. This is shown in the diagram below:

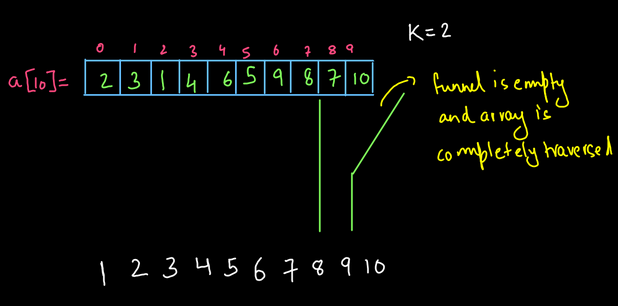
****

We recommend you refer to the solution video to understand the procedure till here as we have skipped the steps in between here as they were the repeating steps.

For more clarity of the question, watch the question video

Play Video

Now, we are in a situation where we do not have any elements left in the array to fill into the funnel. So, we will now keep on removing the elements from the funnel till it becomes empty.

****

Now the funnel is empty and we have also traversed the array completely. So, we get the elements sorted. So, what is the funnel here? Yes, it is the priority queue that we have used to filter out the element of the smallest value every time. We request you to watch the complete procedure once so that everything becomes crystal clear to you. Now let us write the complete code for this procedure:

For more clarity of the question, watch the question video

Play Video

ConsoleJava

import java.io.\*;

import java.util.\*;

public class Main {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

int[] arr = new int[n];

for (int i = 0; i < n; i++) {

arr[i] = Integer.parseInt(br.readLine());

}

int k = Integer.parseInt(br.readLine());

// Add first k+1 elements to the Priority Queue

PriorityQueue< Integer> pq = new PriorityQueue<>();

for (int i = 0; i <= k; i++) {

pq.add(arr[i]);

}

//Filter out the smallest element and move funnel to the next positions

for (int i = k + 1; i < arr.length; i++) {

System.out.println(pq.remove());

pq.add(arr[i]);

}

//Array is completely traversed, empty the funnel now

while (pq.size() > 0) {

System.out.println(pq.remove());

}

}

}

You may refer to the solution video to understand the code written above completely. our video lecture of this problem.

For more clarity of the question, watch the question video

Play Video

Important: Did you observe the first loop in the code? This loop starts from 0 but goes upto k and not k-1. This is because the element at the 0th index can be at three places index 0 which is 0 steps ahead or index 1 which is k-1 steps ahead and index 2 which is k steps ahead (assuming k=2 as in our example). So, we have to store k+1 elements in the priority queue. Now that we have understood this procedure completely, let us analyze the time and space complexity for this code.

**3. Analysis:**

Time Complexity:

The time complexity for this solution is O(n x logk). This is because we are traversing the entire array which takes O(n) time. Inside the traversal loop, we are removing the elements from the priority queue. Removing an element from the priority queue takes O(logk) time since the size of the priority queue is O(k). Hence the overall time complexity becomes O(n x logk).

Space Complexity:

The space complexity for this solution is O(k) as we have created a priority queue of size k+1. Though we are adding and removing the elements from it still we have created it only once and the same priority queue is used for the entire procedure. Thus the space complexity is O(k).

So, dear reader, we hope that you have understood the complete solution. If you have any doubts regarding anything explained above, you may refer to the complete solution video to clear them. With this, we have completed this problem.

For more clarity of the question, watch the question video