PSG COLLEGE OF TECHNOLOGY
DEPARTMENT OF COMPUTER APPLICATIONS
I SEM MCA
23MX17- DATA STRUCTURES LABORATORY
PROBLEM SHEET ON STACKS AND QUEUES

1. A group of students wish to make use of a stack S and a queue Q so as to sort a given sequence of integers, which is initially stored in an array A. At the start, they pushed all the numbers from A to S. The proposed algorithm is as given below:

- Repeat the following steps till the sorted sequence is obtained in A.
- Keeping a track of the maximum and the minimum number, pop all the numbers from S and enqueue them in Q.
- Push all the numbers back to S, except the maximum and the minimum ones.
- The maximum and the minimum numbers have to be placed in A following the required sorted order. No empty space should be there in the array. At the end S and Q become empty and contents of A will give the sorted sequence.

The implementation should handle duplicate numbers and should be able to sort in non-increasing as well as non-decreasing order. Also it should display the contents of array A, stack S (if not empty), and queue Q (if not empty) after completing each of the proposed algorithm iterations.

Input:
Line 1 contains two integers M and N, N is the total number integers to be sorted.
M will decide the sorting order. M=1 represents non-increasing order and M=0 represents non-decreasing order.
Line 2 contains N integers separated by space which are to be sorted.

Output:
The contents of A, S, and Q after each of the iterations should be displayed on a separate line. The contents of an array should be on the first line, next line will have contents of a stack (if not empty), and then comes the contents of a queue (if not empty).

Sample Input: 0 5 26 93 77 44 20
Sample Output: 20 93 26 77 44 20 26 77 93 44 20 26 44 77 93

EXPLANATION:
0 means non-decreasing order.
A[]=26,93,77,44,20
Stack:26,93,77,44,20(top)
Contents of A, S, and Q will vary in the following manner:
I st Iteration 20 is minimum and 93 is maximum.
A[]:20,93
S:44,77,26(top)
Q:EMPTY
II nd Iteration 26 is minimum and 77 is maximum.
A[]:20,26,77,93
S:44(top)
Q:EMPTY
III rd Iteration 44 is minimum and 44 is maximum.
A[]:20,26,44,77,93
S:EMPTY
Q:EMPTY

2. Consider a stack of **N** integers such that the first element represents the top of the stack and the last element represents the bottom of the stack. You need to pop at least one element from the stack. At any one moment, you can convert stack into a queue. The bottom of the stack represents the front of the queue. You cannot convert the queue back into a stack. Your task is to remove exactly **K** elements such that the sum of the **K** removed elements is maximised.

Input format :

- The first line consists of two space-separated integers **N** and **K**.
- The second line consists of **N** space-separated integers denoting the elements of the stack.

Output format :

- Print the maximum possible sum of the **K** removed elements

Sample Input
10 5
10 9 1 2 3 4 5 6 7 8
Sample Output
40
Explanation

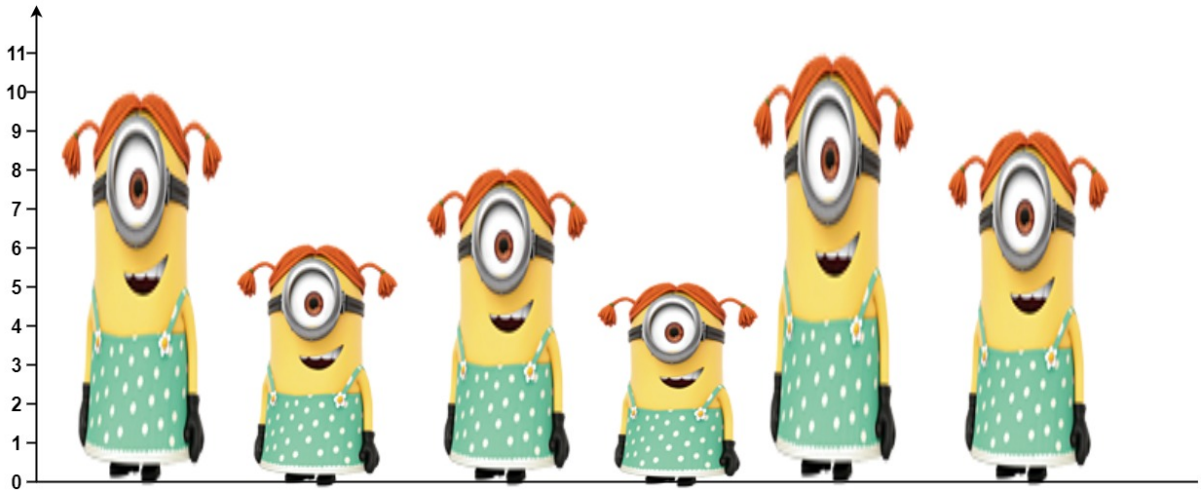
Pop two elements from the stack. i.e {10,9}

Then convert the stack into queue and remove first three elements from the queue. i.e {8,7,6}.

The maximum possible sum is 10+9+8+7+6 = 40

3. There are n people standing in a queue, and they numbered from 0 to n - 1 in left to right order. You are given an array heights of distinct integers where heights[i] represents the height of the i^{th} person.

A person can see another person to their right in the queue if everybody in between is shorter than both of them. More formally, the i^{th} person can see the j^{th} person if i < j and min(heights[i], heights[j]) > max(heights[i+1], heights[i+2], ..., heights[j-1]).
Return an array answer of length n where answer[i] is the number of people the i^{th} person can see to their right in the queue.
Example 1:



Input: heights = [10,6,8,5,11,9]
Output: [3,1,2,1,1,0]
Explanation:
Person 0 can see person 1, 2, and 4.
Person 1 can see person 2.
Person 2 can see person 3 and 4.
Person 3 can see person 4.
Person 4 can see person 5.
Person 5 can see no one since nobody is to the right of them.
Example 2:
Input: heights = [5,1,2,3,10]
Output: [4,1,1,1,0]