ECSE215L: DATA STRUCTURES USING C++ CODATHON - I	
Date: **25-Sep-2021 11:00 AM**	No. of Questions: **1**
Proposed Time: **90 Minutes**	Skills Tested: **Array Insertion, Index tracking with Comparison, Repeatedly finding Minimum in the Array**

ORDER OF REQUEST SERVING

A teacher, Mr. ABC, teaches the subject of XXX to the students of a large classroom. As the course is progressing ahead, he faces a lot of doubts/queries from the students. Due to the large size of the classroom and increasing queries, he often misses the queries of at-risk students (low-scorers). So, to find a solution, he chalks out a strategy to deal with this situation. Below is the strategy:

There will be a simple array of size **n** representing the number of queries that are arriving in a day and a waiting room of size **w** which he uses in his strategy. Each student in the system, is represented by the total marks **m** he/she has scored in the previous semester. The total marks range from 0 to 100.

1. At the start of the day, when the array is empty, the very first student is directly allowed to enter the array using insertion at last (rear end).
2. When the array is non-empty, and next student(s) enter the system, the last (rear end) student marks is checked and compared to the newly entered student.
    a. If the marks of newly entered student is less than or equal to the marks of the student present at rear end, then the student is allowed to enter the array from rear end.
    b. If the marks of newly entered student are more than the marks of the student present at rear end, then the new student is said to wait inside the waiting room.
    c. However, if the capacity of waiting room is exceeded at any point of time (due to the new request) then,
        i. Firstly, the student with minimum marks amongst all the students of the waiting room is picked up and he/she is allowed to enter the array without the rear end marks comparison. In case more than one student is having same minimum marks, then the tie is broken on first come first serve basis.
        ii. Secondly, as now a space is created, the new student is allowed to enter and wait in the waiting room.

    This process is repeated for all further requests.
3. As all the requests are finished, waiting room is checked. Until it is not-empty, one-by-one student with minimum marks is picked up and allowed to enter the array from rear end, without any comparison.

You are requested to output the sequence in which the final requests will be served, by Mr. ABC.

<u>*Input Format:*</u>

w ---> Waiting Room Capacity

n ---> Number of requests arriving in a day

sequence of n student marks with space as separator.

<u>*Output Format:*</u>

Final Sequence of serving the student request.

Number<space>Number<space> ………. Number<space>Number<space>

Input Constraints:

1 <= w <= 50

1 <= n <= 50

0 <= m <= 100

Sample Test Case - 1:

Sample Input:

4   <-- waiting room capacity

8 <-- n: total number of requests in that particular day

10 15 20 25 15 30 5 27 <-- n sequence of marks: [10, 15, 20, 25, 15, 30, 5, 27]

Sample Output:

10 15 5 15 20 25 27 30

Explanation:

For 10: First request, so directly allowed to enter the array

Array:

10							

Waiting Room:

For 15: Compared with 10, as 15>10, so student is said to wait in the waiting room:

Array:

10							

Waiting Room:

15	

For 20: Compared with 10, as 20>10, so student is said to wait in the waiting room:

Array:

10							

Waiting Room:

15	20

For 25: Compared with 10, as 25>10, so student is said to wait in the waiting room:

Array:

10							

Waiting Room:

15	20
25	

For 15: Compared with 10, as 15>10, so student is said to wait in the waiting room:

Array:

10							

Waiting Room:

15	20
25	15

For 30: Compared with 10, as 30>10, so student is said to wait in the waiting room. But waiting room is full, so minimum marks student is picked from the waiting room and is allowed to enter the array without comparison. Here, as there is a tie, so first student with marks 15 is allowed to enter the array. And 30 is said to wait in the waiting room.

Array:

10	15						

Waiting Room:

20	25
15	30

For 5: Compared with 15, as 5<15, so student is directly allowed to enter the array:

Array:

10	15	5					

Waiting Room:

20	25
15	30

For 27: Compared with 5, as 27>5, so student is said to wait in the waiting room. But again waiting room is full so, next minimum candidate is picked and entered into the array. And 27 is said to wait in the waiting room.

Array:

10	15	5	15				

Waiting Room:

20	25
30	27

Now, since all the requests have finished, so now waiting room is checked and one-by-one minimum marks student is picked and entered in the array.

Array:

10	15	5	15	20	25	27	30

Waiting Room:

So, the final sequence of request served is: [10, 15, 5, 15, 20, 25, 27, 30]

<u>Sample Test Case - 2:</u>

Sample Input:

3 <-- waiting room capacity

6 <-- n: total number of requests in that particular day

96 49 31 62 45 87 <-- n sequence of marks: [96, 49, 31, 62, 45, 87]

Sample Output:

96 49 31 45 62 87

Explanation:

For 96: First request, so directly allowed to enter the array.

Array:

96					

Waiting Room:

For 49: Compared with 96, as 49<96, so allow student to enter the array:

Array:

96	49				

Waiting Room:

For 31: Compared with 49, as 31<49, so student is allowed to enter the array:

Array:

96	49	31			

Waiting Room:

For 62: Compared with 31, as 62>31, so student is said to wait in the waiting room:

Array:

96	49	31			

Waiting Room:

62		

For 45: Compared with 31, as 45>31, so student is said to wait in the waiting room:

Array:

96	49	31			

Waiting Room:

62	45	

For 87: Compared with 31, as 87>31, so student is said to wait in the waiting room:

Array:

96	49	31			

Waiting Room:

62	45	87

Now, since all the requests have finished, so now waiting room is checked and one-by-one minimum marks student is picked and entered in the array.

Array:

96	49	31	45	62	87

Waiting Room:

So, the final sequence of request served is: [96, 49, 31, 45, 62, 87]

TEST CASES:

Test Case No.	**Input**	**Output**

Sample Test Case - 1
(As per the question example)
	4
	8
	10 15 20 25 15 30 5 27
		10 15 5 15 20 25 27 30

Sample Test Case – 2
(As per the question example)
	3
	6
	96 49 31 62 45 87
		96 49 31 45 62 87

Marked Test Case 1	5
	10
	50 40 30 20 10 90 60 70 80 75
		50 40 30 20 10 60 70 75 80 90

Marked Test Case 2	2
	6
	15 25 10 30 30 30
		15 10 25 30 30 30

Marked Test Case 3	2
	6
	15 10 5 20 14 6
		15 10 5 14 6 20

Marked Test Case 4	1
	6
	35 45 30 55 40 60
		35 30 45 40 55 60

Marked Test Case 5	2
	6
	50 60 50 48 47 46
		50 50 48 47 46 60

Marked Test Case 6	2
	8
	60 61 62 63 64 65 66 67
		60 61 62 63 64 65 66 67

Marked Test Case 7	1 1 100	100

Marked Test Case 8	50
	50
	47 97 55 100 44 97 26 92 78 97 5 16 33 33 44 17 61 98 4 79 21 9 20 37 49 100 45 52 67 61 74 0 92 87 77 25 16 47 95 63 88 22 26 5 92 10 98 85 38 35
		47 44 26 5 4 0 5 9 10 16 16 17 20 21 22 25 26 33 33 35 37 38 44 45 47 49 52 55 61 61 63 67 74 77 78 79 85 87 88 92 92 92 95 97 97 97 98 98 100 100

Marked Test Case 9	1
	50
	47 97 55 100 44 97 26 92 78 97 5 16 33 33 44 17 61 98 4 79 21 9 20 37 49 100 45 52 67 61 74 0 92 87 77 25 16 47 95 63 88 22 26 5 92 10 98 85 38 35
		47 97 55 44 100 26 97 78 92 5 97 33 33 16 44 17 61 4 98 21 9 79 37 20 49 45 100 67 61 52 0 74 92 77 25 16 87 47 95 88 22 63 5 26 10 92 85 38 35 98

Marked Test Case 10	5
	50
	97 47 55 100 44 97 26 92 78 97 5 16 33 33 44 17 61 89 4 79 21 9 20 37 49 100 45 52 67 62 74 0 92 87 77 25 16 47 95 63 88 22 26 5 92 5 98 85 38 29
		97 47 44 26 55 5 78 33 33 16 44 17 61 4 89 21 9 79 37 20 49 45 92 67 62 52 0 74 92 77 25 16 87 47 95 88 22 63 5 26 5 92 85 38 29 97 97 98 100 100

Proposed Solution Code:

```
#include <iostream>
using namespace std;
int find_and_remove_min(int waiting[],int w)
{
    int min=0,val,i;
    for(i=1;i<w;i++)
    {
        if(waiting[i]<waiting[min])
            min=i;
    }
    val=waiting[min];
    for(i=min;i<w-1;i++)
    {
        waiting[i]=waiting[i+1];
    }
    waiting[i]=1000;
    return val;
}
int main() {
    int w,n,i,j;
    cin>>w>>n;
    int input[n];
    int waiting[w];
    int request[n];
    for(i=0;i<n;i++)
            cin>>input[i];
    for(j=0;j<w;j++)
        waiting[j]=1000;
    int rear=0;
    int w_index=0;
    for(i=0;i<n;i++)
    {
        if(i==0 && rear==0)
        {
            request[rear++]=input[i];
        }
        else
        {
            if(request[rear-1]>=input[i])
            {
                request[rear++]=input[i];
            }
            else
            {
                if(w_index<w)
                {
                    waiting[w_index++]=input[i];
                }
                else
                {
                    request[rear++]=find_and_remove_min(waiting,w);
                    waiting[w_index-1]=input[i];
                }
            }
        }
    }
    for(j=0;j<w;j++)
    {
            int min=0,val,i;
            for(i=1;i<w;i++)
            {
                if(waiting[i]<waiting[min])
                min=i;
            }
            val=waiting[min];
            waiting[min]=1000;
            request[rear++]=val;
    }
    for(j=0;j<n;j++)
    {
        cout<<request[j]<<" ";
    }
    return 0;
}
```