Assignment 3 :- You have a queue of integers that you need to sort. You can only use additional space equivalent to one stack. Describe the steps you would take to sort the elements in the queue.

1.Initialization:

Create an empty stack to assist in the sorting process.

2.Find and Move the Minimum Element:

Repeatedly find the minimum element in the queue and move elements to the stack and back to the queue to maintain the order and find the next minimum element.

3.Store Sorted Elements:

Once the minimum element is identified, it is moved to its correct position.

Here are the detailed steps:

Steps to Sort the Queue

1.Initialization:

Let queue be the input queue.

Create an empty stack.

2.Outer Loop:

Repeat the process until the queue is sorted (i.e., the queue size equals the original size and elements are in non-decreasing order).

3.Find Minimum and Move Elements:

Initialize a variable min to store the minimum value found.

Determine the size of the queue at the start of each iteration to ensure proper control over the number of elements processed.

Traverse the queue to find the minimum element while using the stack to temporarily hold elements:

Dequeue each element from the queue.

Compare it with min, and update min if a smaller element is found.

Push each dequeued element onto the stack.

4.Rebuild Queue and Move Minimum Element to the End:

Transfer elements back to the queue from the stack, except for the minimum element, which should be pushed onto the stack at the correct position.

5.Preserve Sorted Elements:

Move the minimum element to the end of the queue (or beginning if you're implementing in-place queue sorting).

6.Repeat:

Continue the process until all elements are sorted.

Assignment 5:- A sorted linked list has been constructed with repeated elements. Describe an algorithm to remove all duplicates from the linked list efficiently.

1.Initialization:

Start with the head of the list.

Use a pointer, say current, to traverse the list.

2.Traverse the List:

For each node, check if it has the same value as the next node.

If it does, bypass the next node by setting current.next to current.next.next.

If it doesn't, move the current pointer to the next node.

3.Repeat Until End:

Continue this process until you reach the end of the list.