Task3 : Queue Sorting with Limited Space

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.

To sort a queue of integers using limited space equivalent to one stack, you can follow these steps:

1. Initialize an empty stack.
2. While the queue is not empty: a. Dequeue an element from the queue. b. If the stack is empty or the dequeued element is greater than the top element of the stack, push the dequeued element onto the stack. c. If the dequeued element is smaller than the top element of the stack, keep popping elements from the stack and enqueue them back into the queue until you find the correct position for the dequeued element in the stack. Then, push the dequeued element onto the stack.
3. After all elements have been processed, the stack will contain the sorted elements in descending order. You can reverse the order by dequeuing elements from the stack and enqueueing them back into the queue.
4. Now, the queue will contain the sorted elements in ascending order.

```
package QueueSorting;

import java.util.Queue;
import java.util.Stack;

public class queueSorting {
        public static void sortQueue(Queue<Integer> queue) {
            Stack<Integer> stack = new Stack<>();

            while (!queue.isEmpty()) {
                int current = queue.poll();

                while (!stack.isEmpty() && stack.peek() > current) {
                    queue.offer(stack.pop());
                }

                stack.push(current);
            }

            while (!stack.isEmpty()) {
                queue.offer(stack.pop());
            }
        }
}
```

Main Method -

```
public class queueSortMain {
        public static void main(String[] args) {
          Queue<Integer> queue = new LinkedList<>();
          queueSorting q = new queueSorting();
          queue.offer(5);
          queue.offer(3);
          queue.offer(8);
          queue.offer(1);
          queue.offer(2);

          System.out.println("Queue before sorting: " + queue);

          q.sortQueue(queue);

          System.out.println("Queue after sorting: " + queue);
        }
}
```