Stack Using One Queue LeetCode

This C++ program demonstrates the implementation of a stack using a single **queue** container. The **Stack** class is defined with a private member variable **q**, which is a **queue<int>**. The program simulates the stack behavior by manipulating the elements within the queue.

The **Stack** class is defined with a private member variable **q**, which is a **queue<int>** container.

1. **push (void push(int value)):**
   - Function Explanation: Adds the given element to the top of the stack by pushing it onto the queue and then rotating the queue to maintain the stack order.
   - **Time Complexity: O(n), where n is the number of elements in the stack. The rotation operation involves moving the front element to the back q.size() - 1 times.**
   - **Space Complexity: O(1)**
2. **pop (int pop()):**
   - Function Explanation: Removes and returns the top element from the stack (the front element of the queue).
   - **Time Complexity: O(1)**
   - **Space Complexity: O(1)**
3. **getTop (int getTop()):**
   - Function Explanation: Returns the top element of the stack (the front element of the queue) without removing it.
   - **Time Complexity: O(1)**
   - **Space Complexity: O(1)**
4. **isEmpty (bool isEmpty()):**
   - Function Explanation: Checks if the stack is empty by examining whether the **queue** container is empty.
   - **Time Complexity: O(1)**
   - **Space Complexity: O(1)**
5. **getSize (int getSize()):**

- Function Explanation: Returns the number of elements in the stack, which is the size of the **queue** container.
- **Time Complexity: O(1)**
- **Space Complexity: O(1)**

6. **Destructor (~Stack()):**
    - Function Explanation: Releases memory used by the **queue** by iteratively popping elements from it.
    - **Time Complexity: O(n), where n is the number of elements in the stack.**
    - **Space Complexity: O(1)**