|  |  |
| --- | --- |
| Reverse k elements in C++ | |
| #include <iostream>  #include <queue>  #include <stack>  using namespace std;  queue<int> modifyQueue(queue<int> q, int k) {  stack<int> st;  // Push the first k elements into a stack  for (int i = 0; i < k; i++) {  st.push(q.front());  q.pop();  }  // Pop elements from the stack and enqueue them back into the queue  while (!st.empty()) {  q.push(st.top());  st.pop();  }  // Rotate the remaining elements in the queue  int size = q.size();  for (int i = 0; i < size - k; i++) {  q.push(q.front());  q.pop();  }  return q;  }  int main() {  // Create a queue and add some elements  queue<int> q;  q.push(1);  q.push(2);  q.push(3);  q.push(4);  q.push(5);  // Define the value of k  int k = 3;  // Call the modifyQueue function and store the result  queue<int> result = modifyQueue(q, k);  // Print the result queue  while (!result.empty()) {  cout << result.front() << " ";  result.pop();  }  cout << endl;  return 0;  } | Step-by-Step Execution  Step 1: Push first k elements into a stack   | Operation | Stack (Top to Bottom) | Queue | | --- | --- | --- | | push 1 | 1 | [2, 3, 4, 5] | | push 2 | 2, 1 | [3, 4, 5] | | push 3 | 3, 2, 1 | [4, 5] |   Step 2: Pop from stack and enqueue back   | Operation | Stack | Queue | | --- | --- | --- | | pop 3 | 2, 1 | [4, 5, 3] | | pop 2 | 1 | [4, 5, 3, 2] | | pop 1 | empty | [4, 5, 3, 2, 1] |   Step 3: Rotate the remaining size - k elements (5 - 3 = 2 times)   | Operation | Queue before | Queue after | | --- | --- | --- | | move 4 | [4, 5, 3, 2, 1] | [5, 3, 2, 1, 4] | | move 5 | [5, 3, 2, 1, 4] | [3, 2, 1, 4, 5] |   ✅ Final Queue:  [3, 2, 1, 4, 5]  📤 Output:  3 2 1 4 5 |
| 3 2 1 4 5 | |