Пермский национальный исследовательский политехнический университет.

Лабораторная работа № 11.

«Информационные динамические структуры».

Выполнил: студент группы РИС-23-2б

Вековшинин Иван Николаевич

Проверила: доцент кафедры ИТАС

О.А. Полякова.

2024 г.

Постановка задачи:

Написать программу, в которой создаются динамические структуры и выполнить их обработку в соответствии со своим вариантом.

Записи в линейном списке содержат ключевое поле типа \*char(строка символов). Сформировать двунаправленный список. Удалить элемент с заданным ключом. Добавить К элементов перед элементом с заданным номером.

Решение с помощью списков.

Анализ задачи:

1. Создать структуру Node с полями key, next и prev.

2. Создать функцию createNode для создания нового узла с заданным ключом.

3. Создать функцию addNodesBefore для добавления K элементов перед элементом с заданным номером.

4. Создать функцию deleteNode для удаления элемента с заданным ключом.

5. Создать функцию printList для вывода элементов списка.

6. В функции main:

6.1. Инициализировать указатель head как nullptr.

6.2. Ввести количество элементов n, которые нужно добавить в список.

6.3. Для каждого элемента:

- Ввести ключ элемента.

- Создать новый узел с этим ключом и добавить его в конец списка.

6.4. Вывести исходный список.

6.5. Ввести ключ элемента, который нужно удалить.

6.6. Удалить элемент с этим ключом.

6.7. Вывести список после удаления элемента.

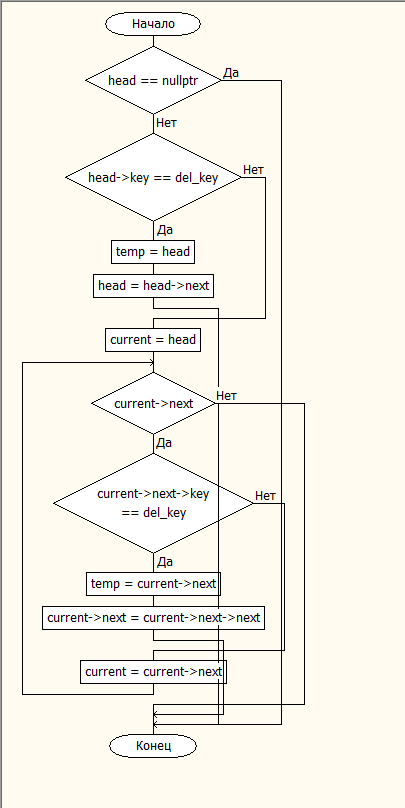
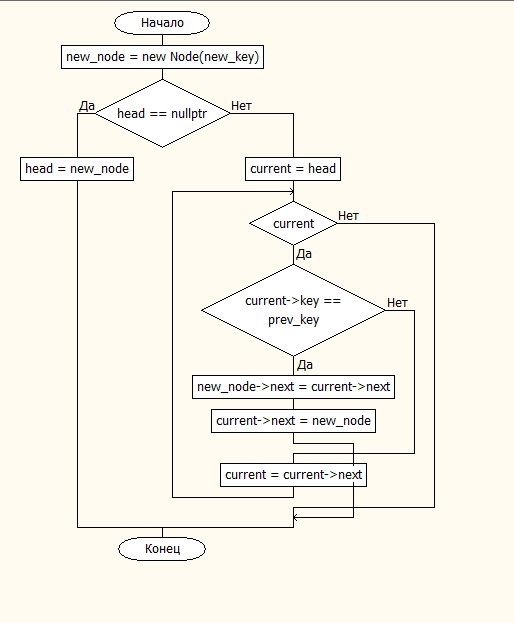
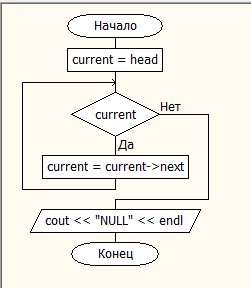
6.8. Ввести номер элемента и количество элементов K, которые нужно добавить перед ним.

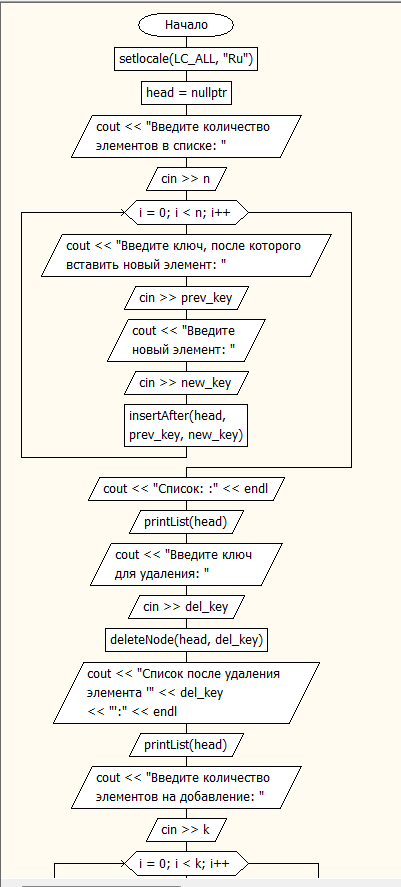
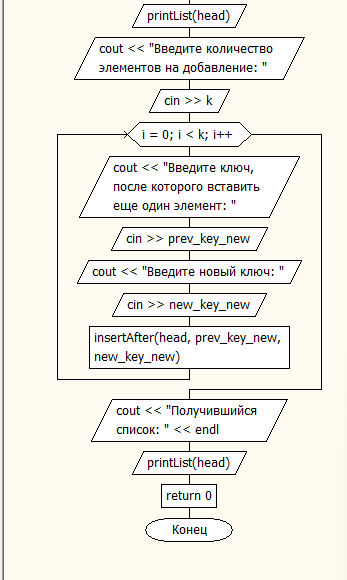
6.9. Добавить K элементов перед указанным номером.

6.10. Вывести список после добавления элементов.

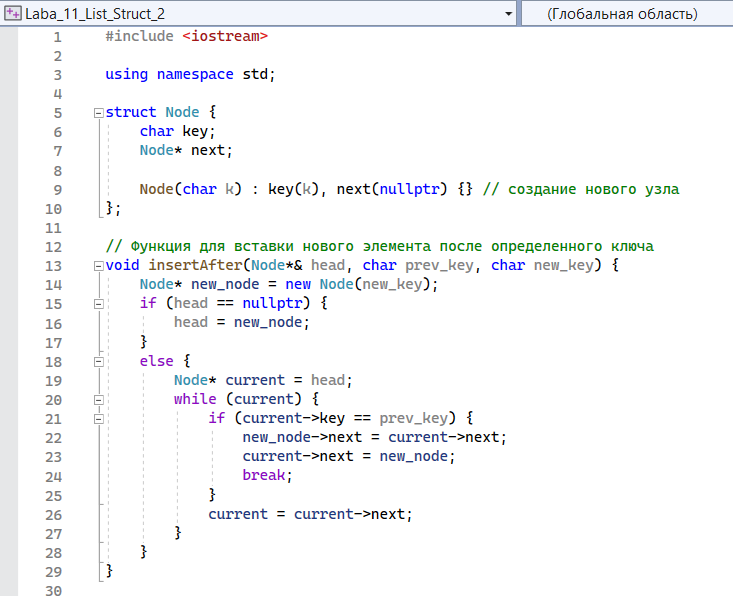
1. **Решение односвязным списком**

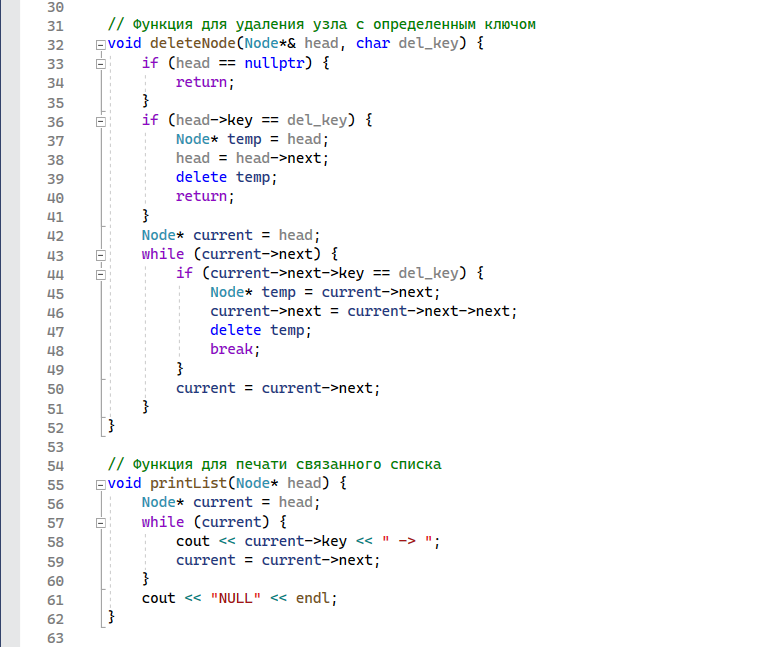
Решение блок схемой:

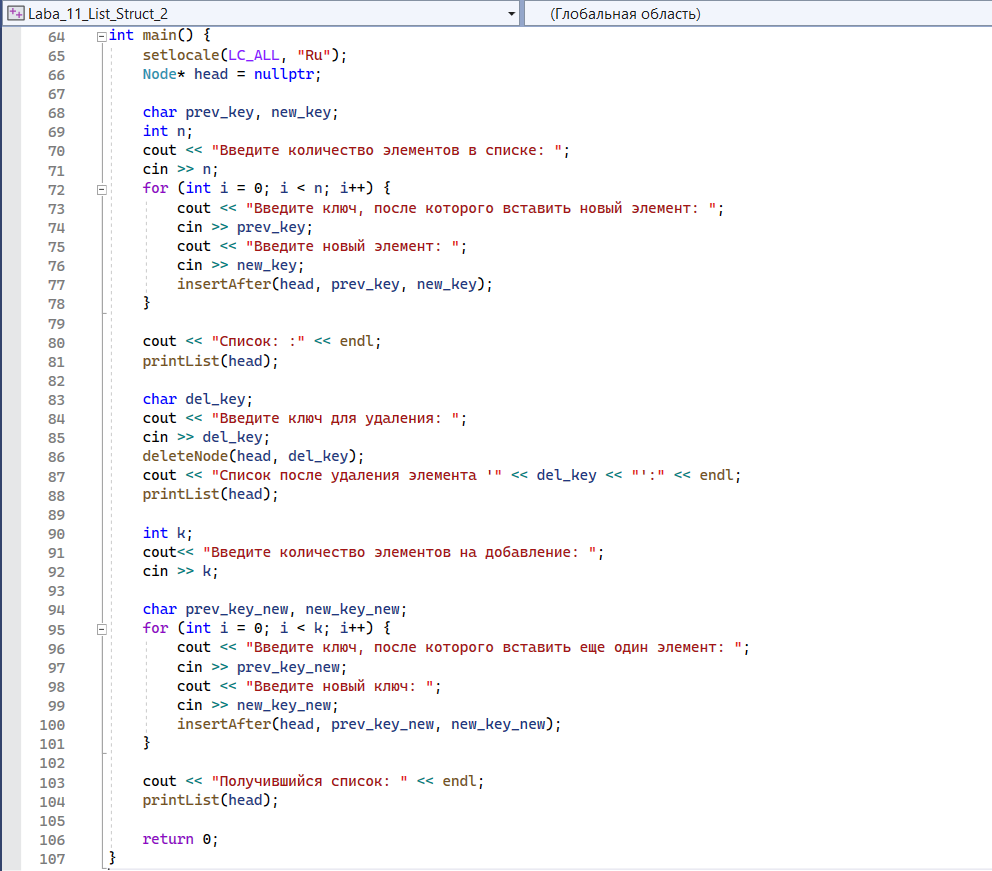
  

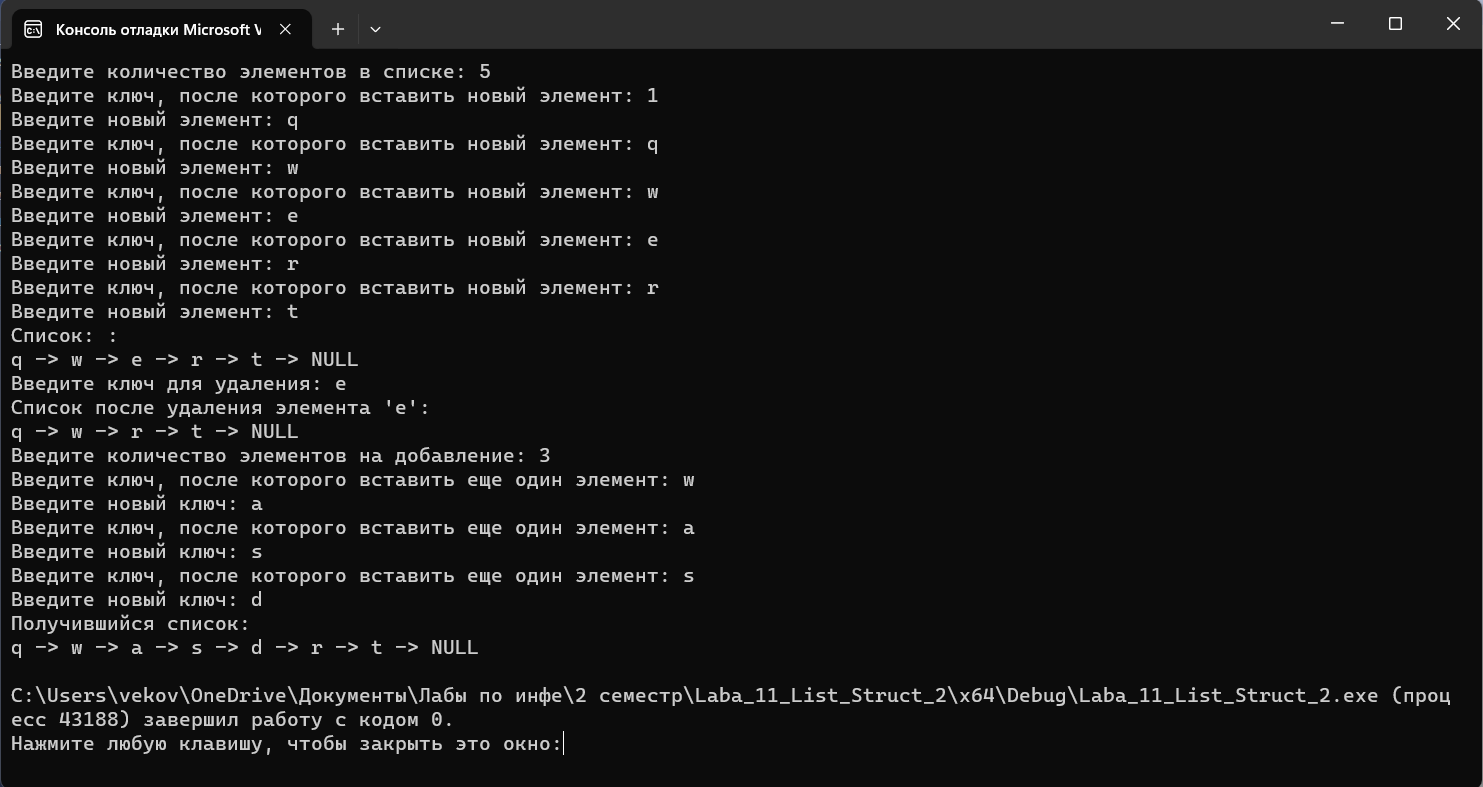
 

Решение кодом:



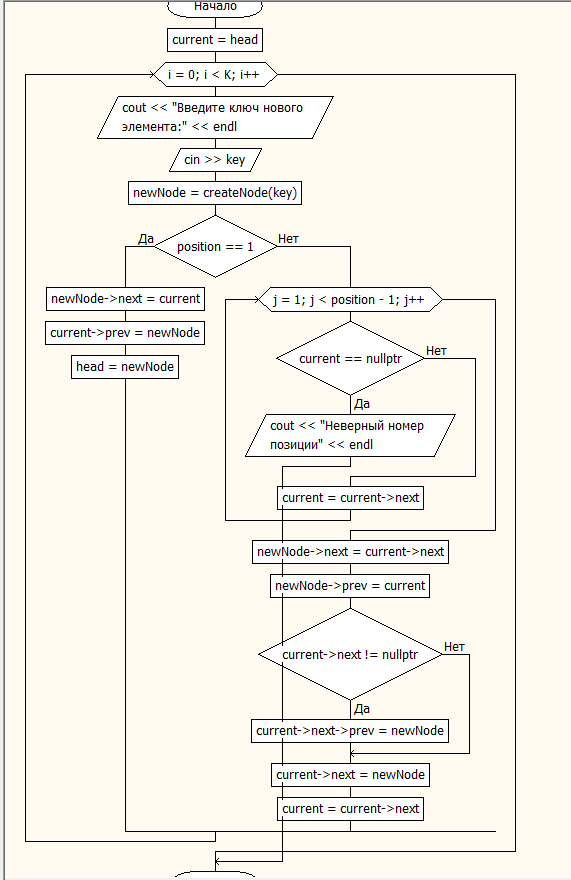
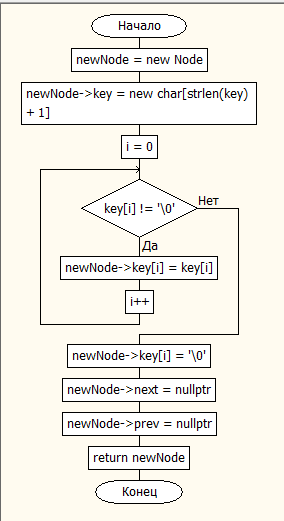
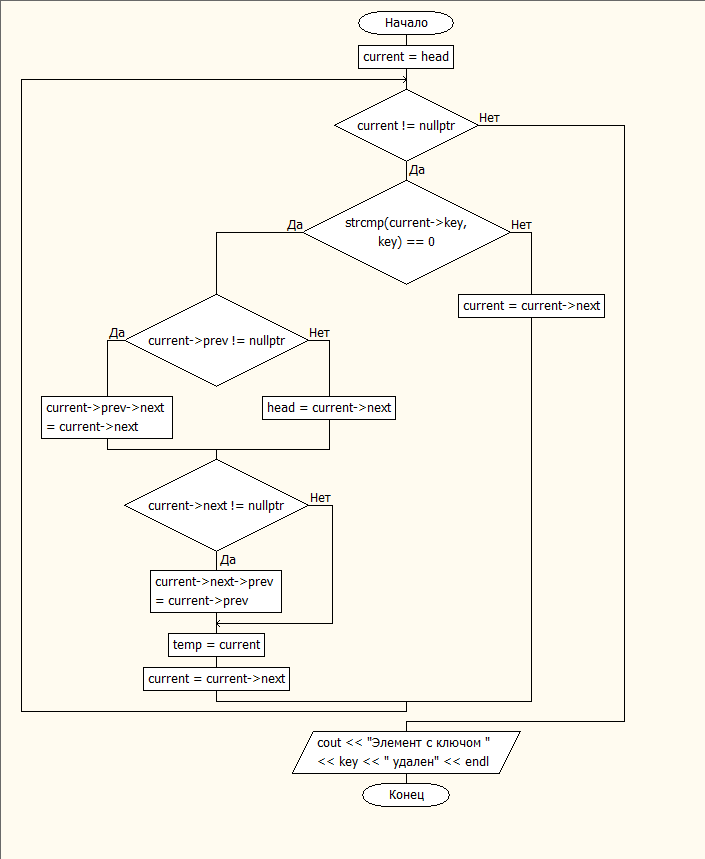
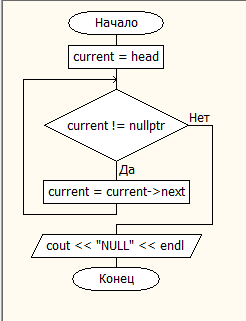
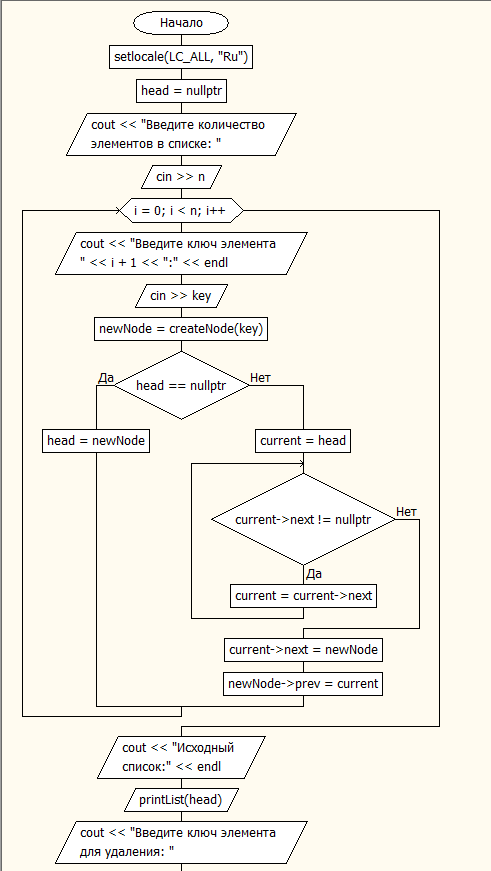
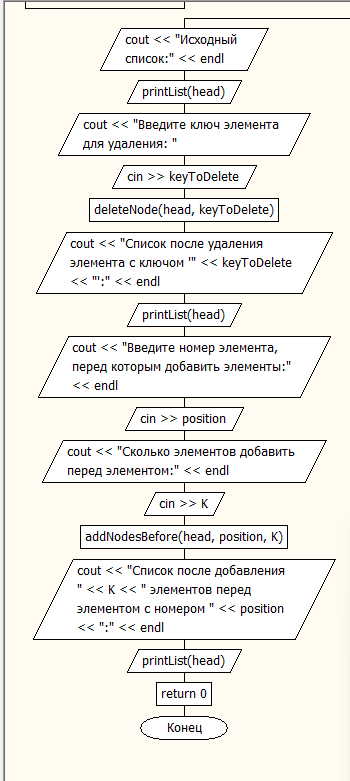




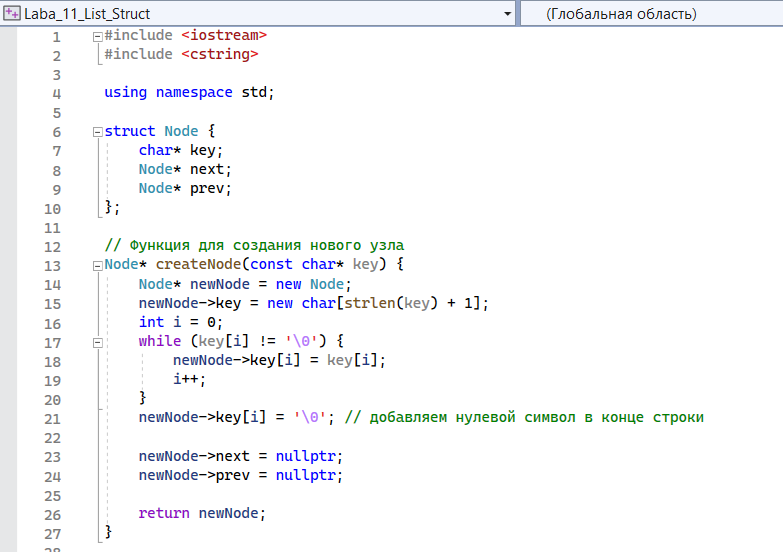


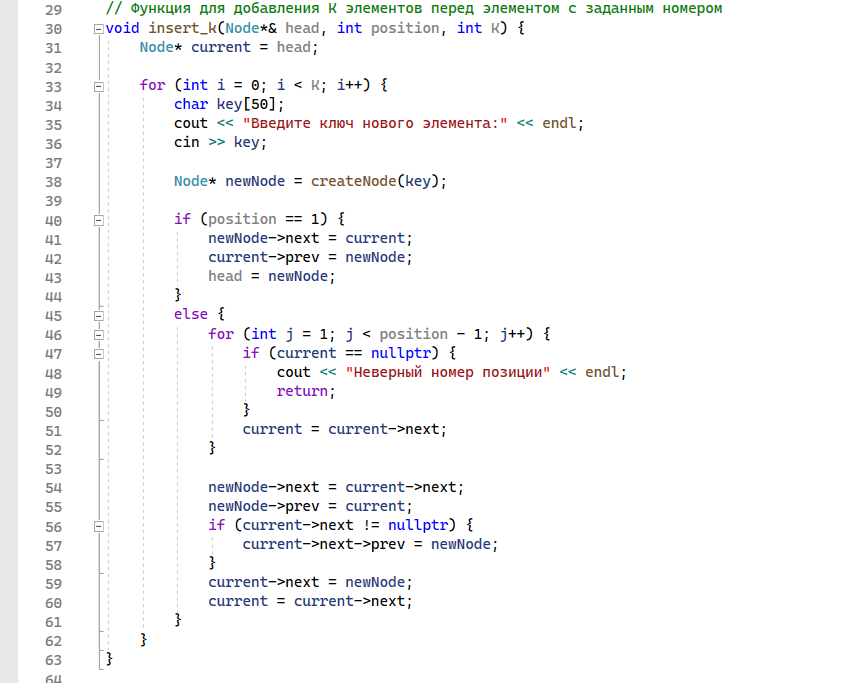
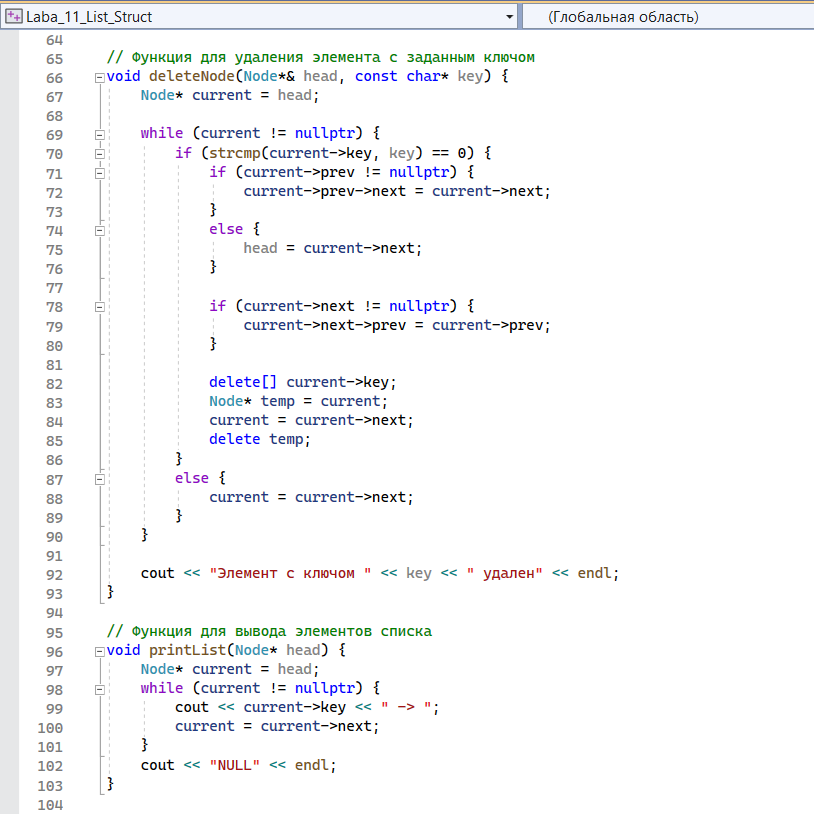
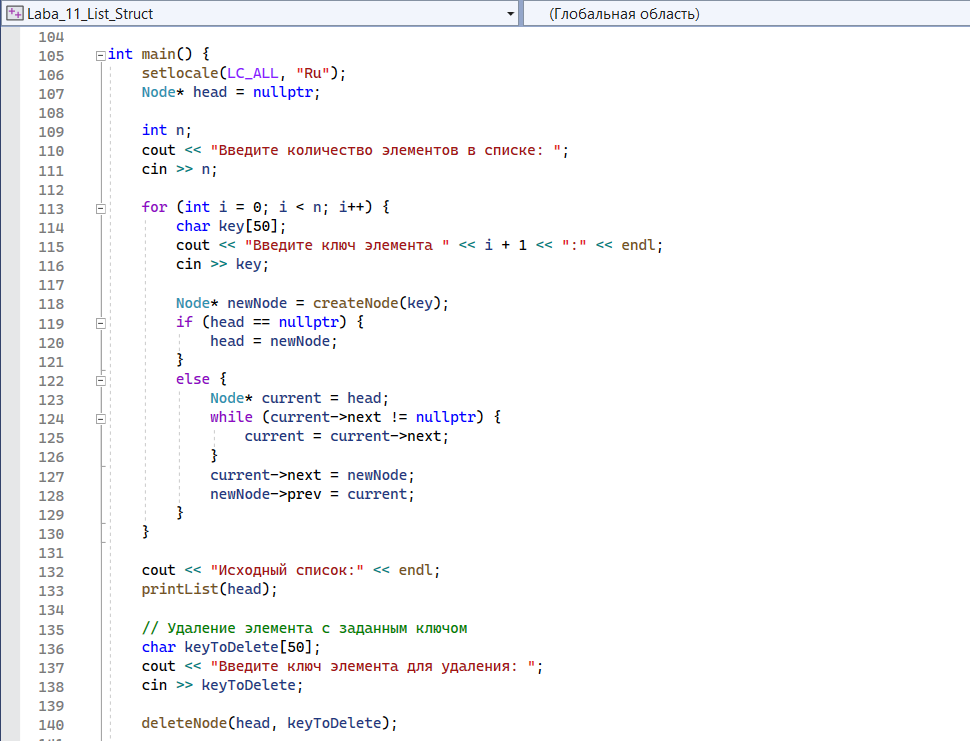
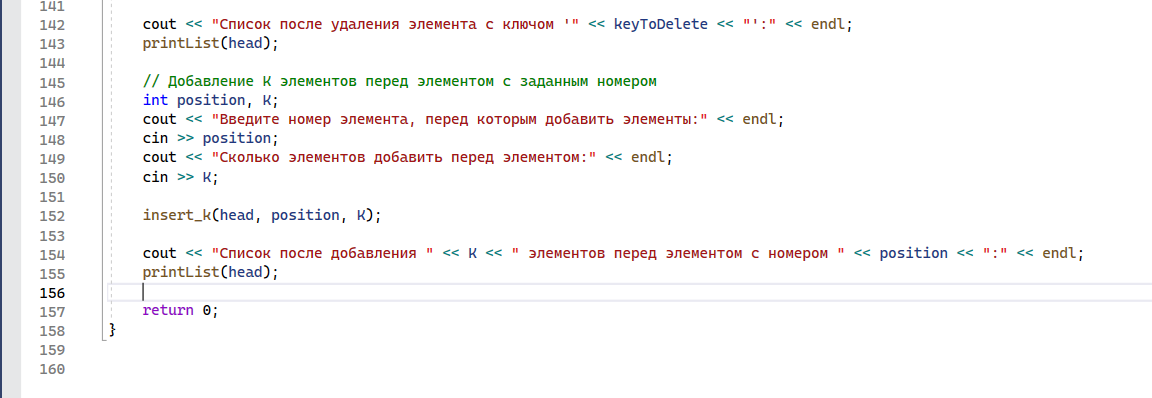
1. **Решение двусвязным списком**

Решение блок схемой:

Решение кодом:

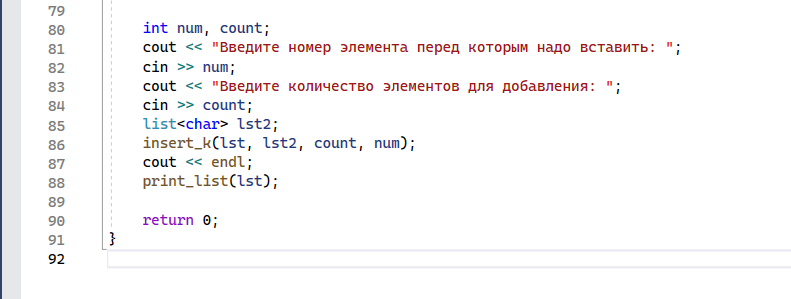
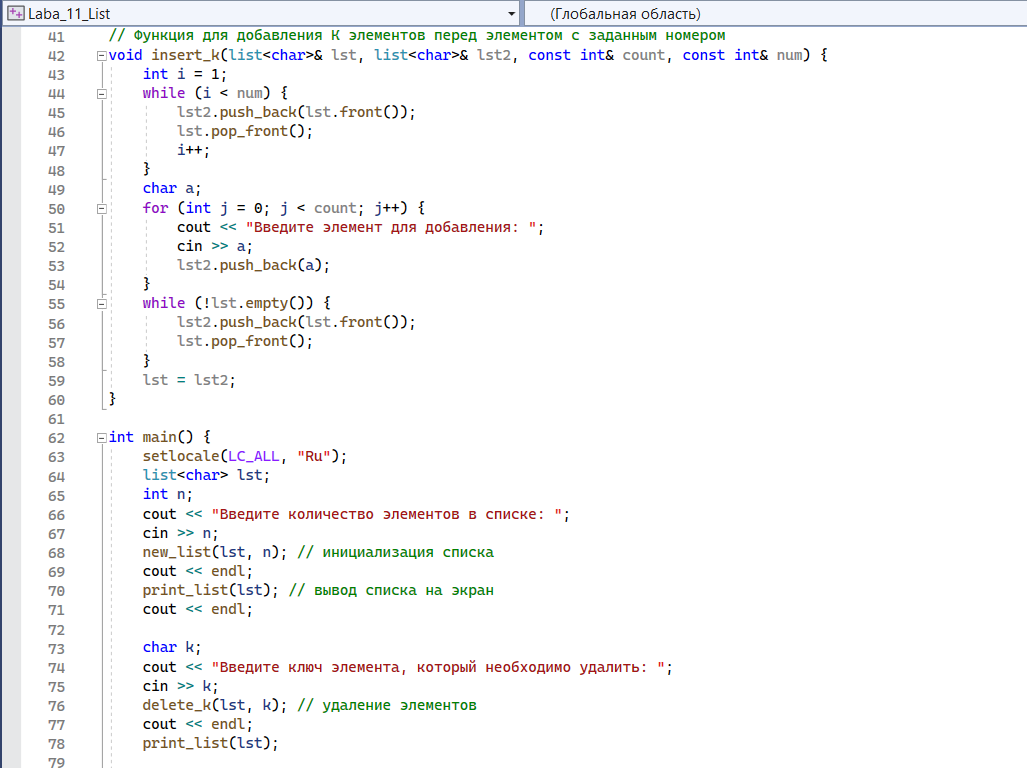
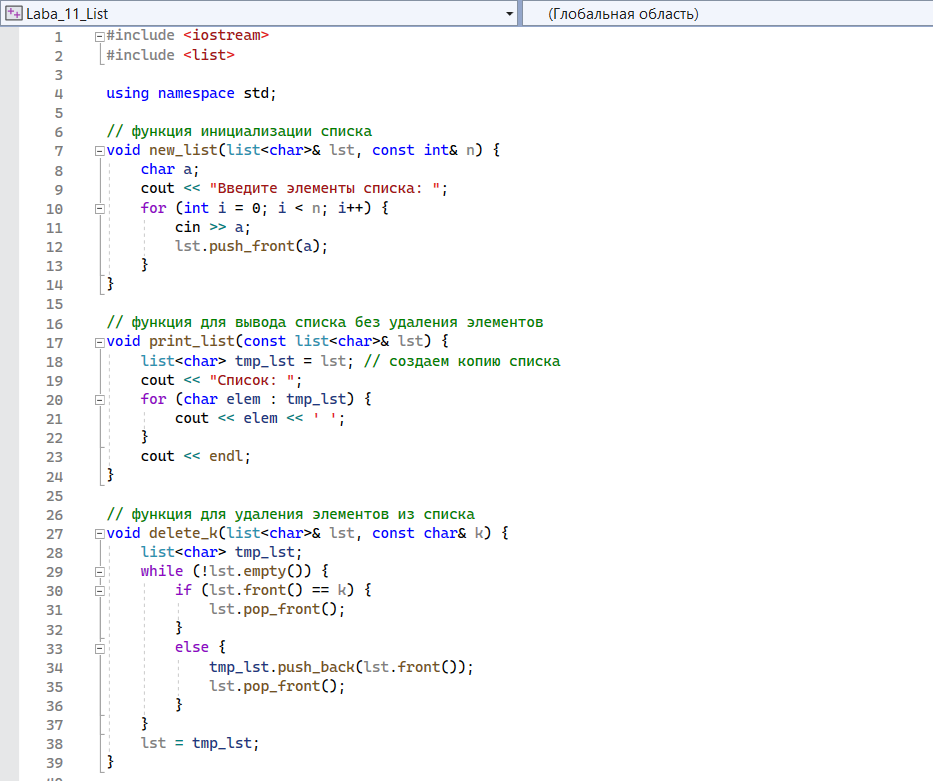


1. **Решение с помощью библиотеки List.**

Решение блок схемой:

Решение кодом:

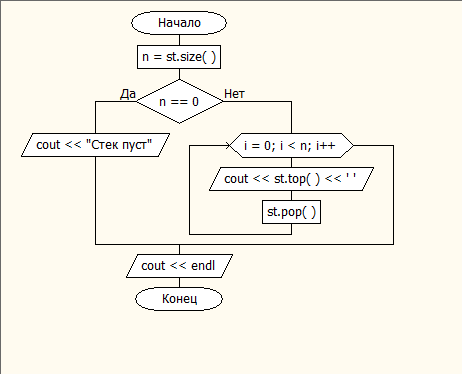
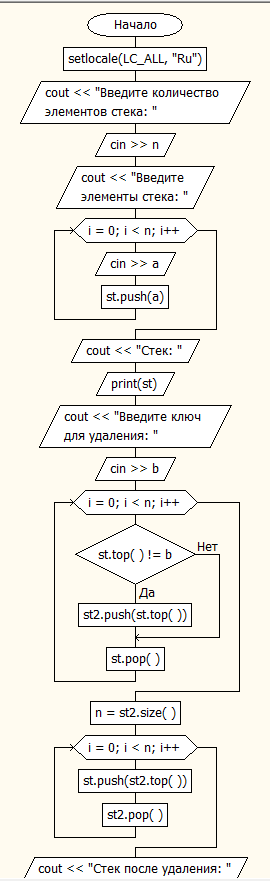
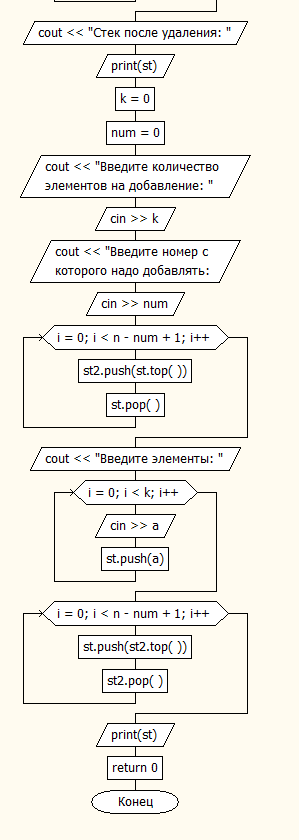


**Решение с помощью стеков.**

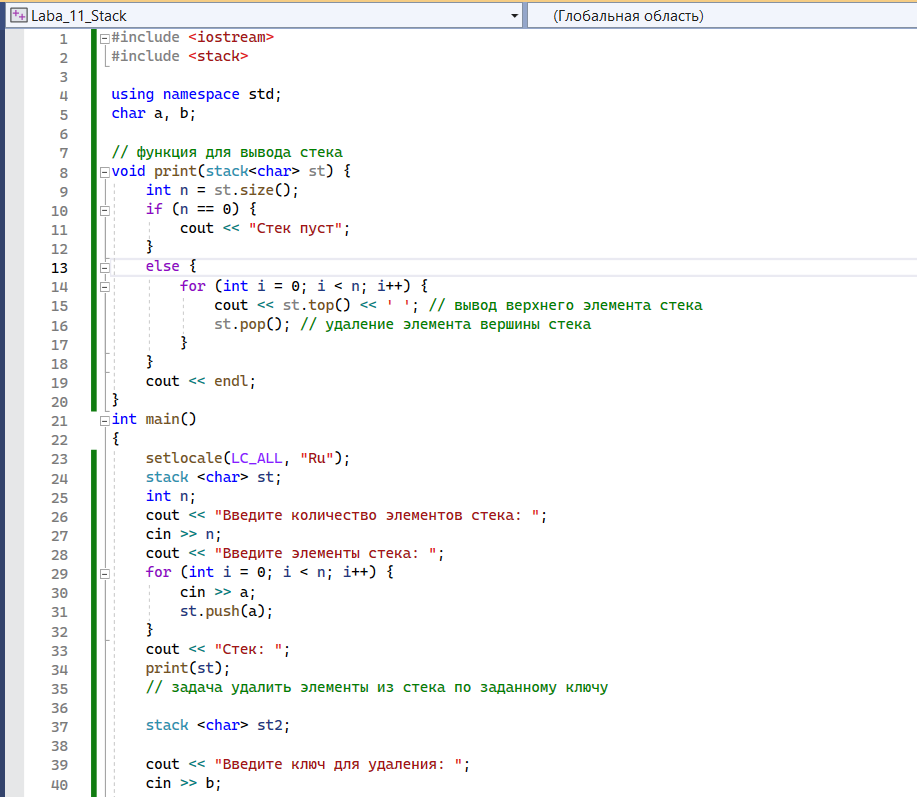
Анализ задачи:

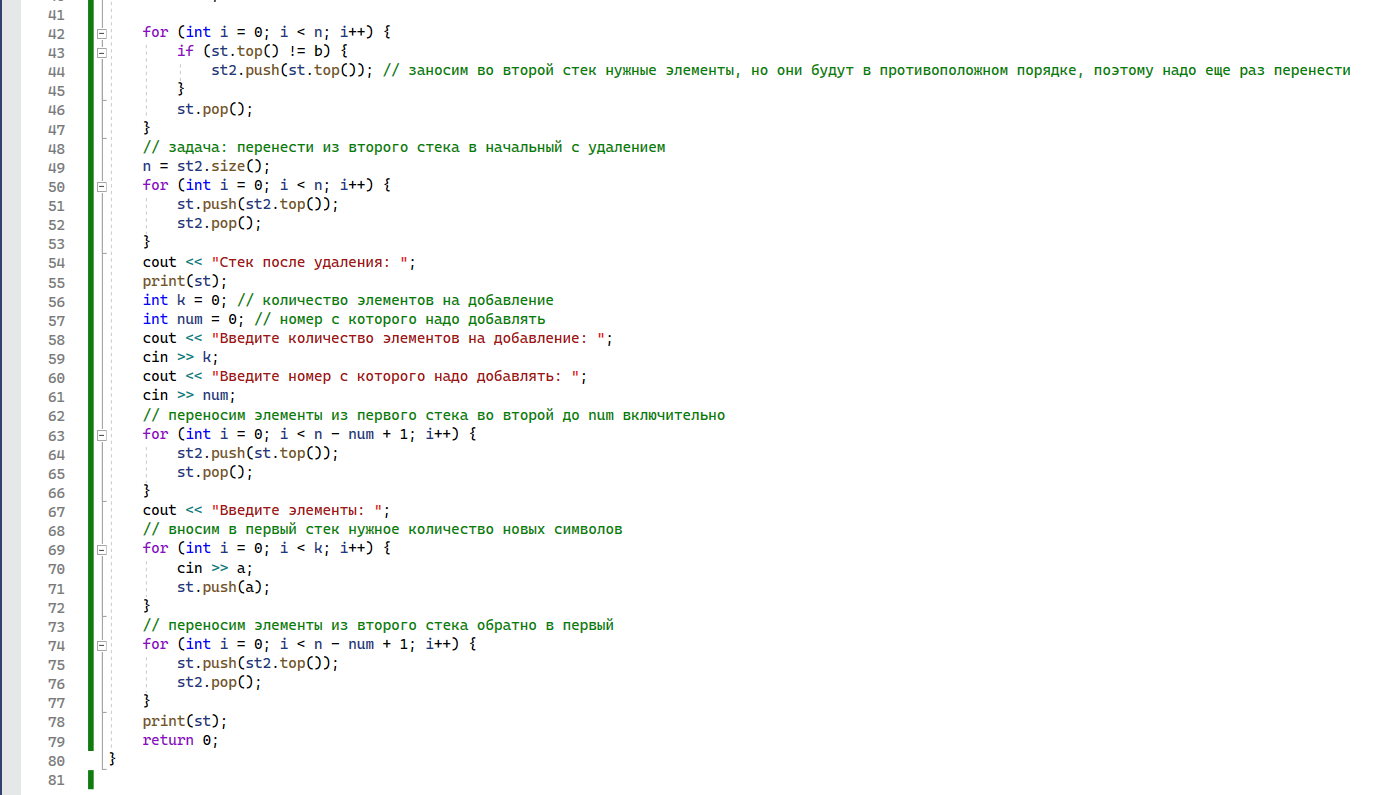
1. Для реализации программы через STL-библиотеку необходимо подключить библиотеку <stack>.
2. Из условий задачи следует, что необходимо выводить стек несколько раз. Чтобы избежать повторений кода, создадим функцию void print(stack<char> st) для вывода элементов стека. Введем переменную int p, которая будет содержать размер стека с помощью size(). Затем с помощью условного оператора if проверим, пуст ли стек или нет. Если стек пуст, выведем сообщение о его пустоте, иначе с помощью цикла for (int i = 0; i < p; i++) будем выводить элементы стека и удалять их после вывода вершины стека.
3. Записи в стеке содержат ключевое поле типа char (строка символов). Для формирования стека необходимо создать его (stack<char> st;). Пользователь должен ввести количество элементов в стеке (int n). Затем с помощью цикла for (int i = 0; i < n; i++) введем элементы стека в переменную char a и добавим их в стек. После этого с помощью функции print выведем элементы стека на экран.
4. Для удаления элемента с заданным ключом можно создать второй стек (stack<int> st2), в который будут перенесены нужные элементы из основного стека. Пользователь должен ввести ключ (b) для удаления. Затем с помощью цикла for и проверки if (st.top() != b) будем переносить нужные элементы во второй стек (st2.push(st.top())). Для определения количества элементов после удаления можно использовать size (n = st2.size()). Затем с помощью цикла for перенесем элементы из второго стека обратно в основной. Затем можно использовать функцию print для вывода элементов стека.
5. Для добавления K элементов перед элементом с заданным номером можно попросить пользователя ввести K и номер элемента s, перед которым будут добавлены элементы. Затем перенесем нужные элементы во второй стек с помощью цикла for (int i = 0; i < n-s+1; i++) (n-s+1, чтобы включить элемент s в переносимые). Затем с помощью цикла for добавим K элементов в основной стек. После этого с помощью нового цикла перенесем элементы из второго стека обратно в основной и снова воспользуемся функцией print для вывода элементов стека.
6. **Решение с помощью библиотеки <stack>**

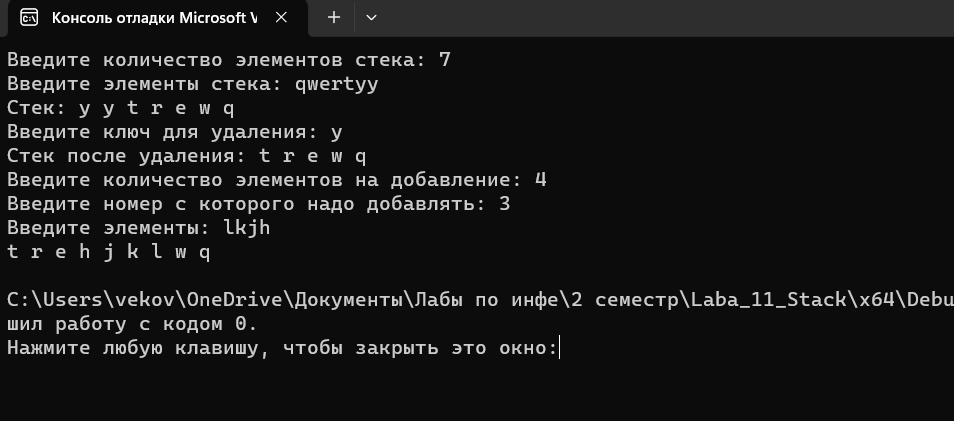
Решение блок схемой:

Решение кодом:



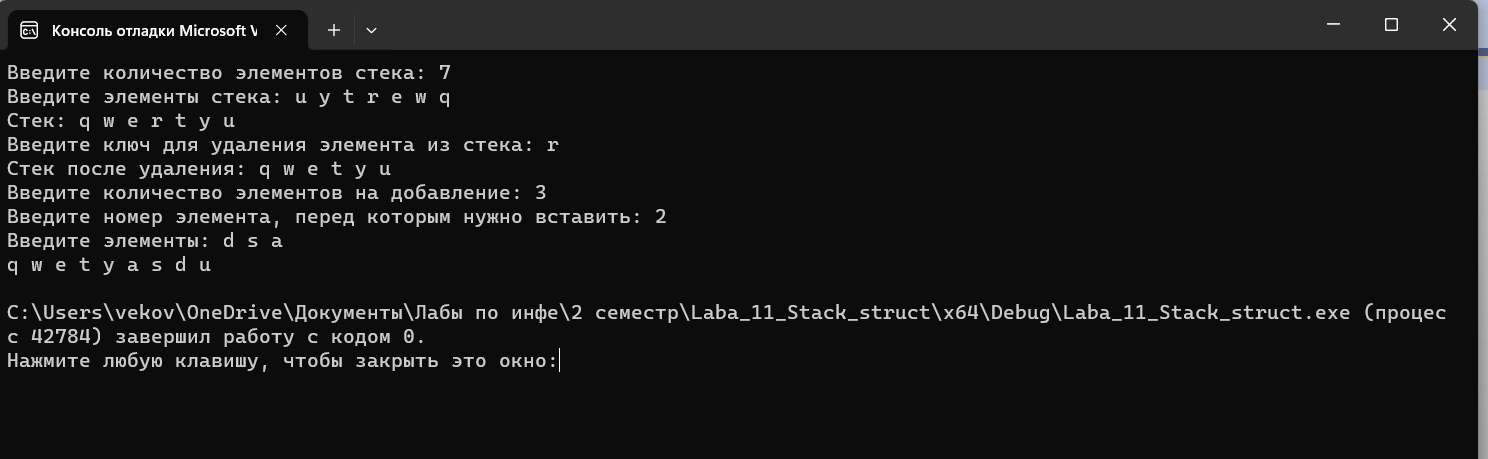
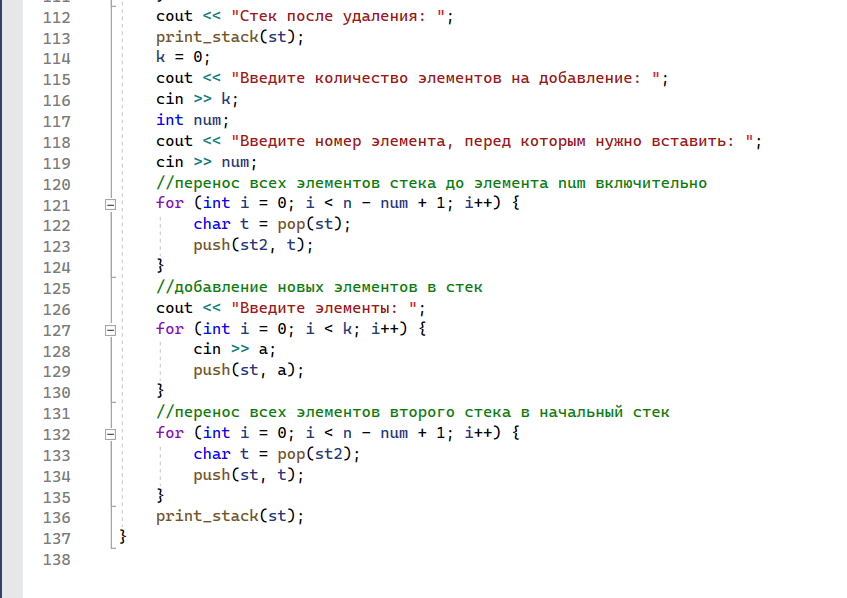
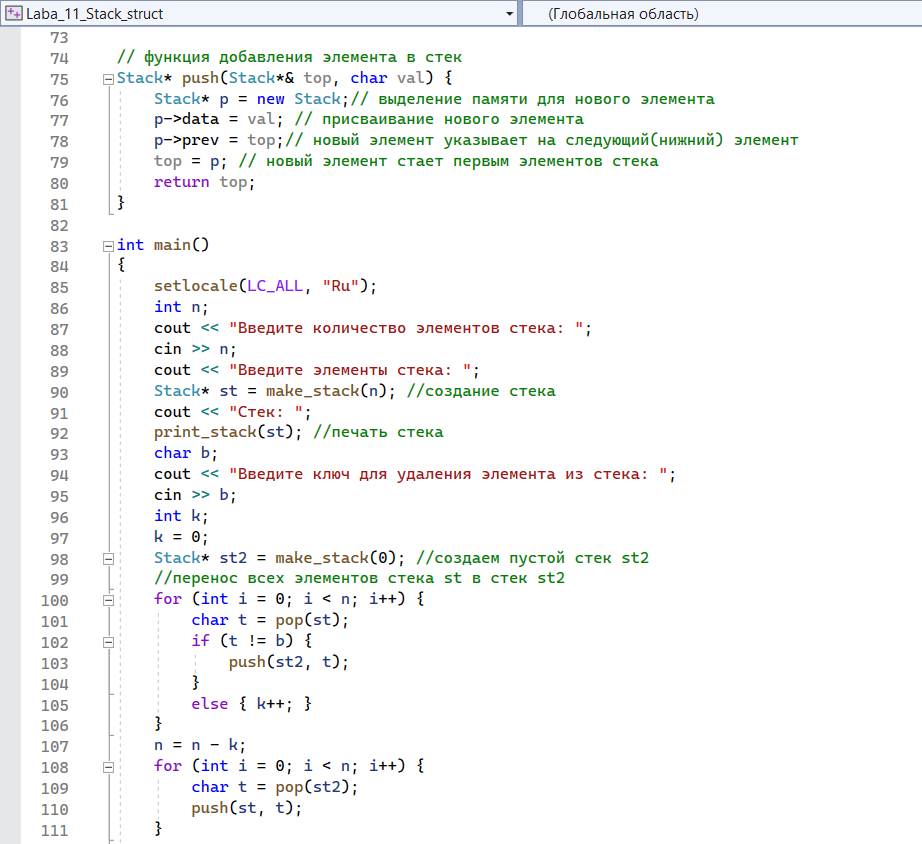
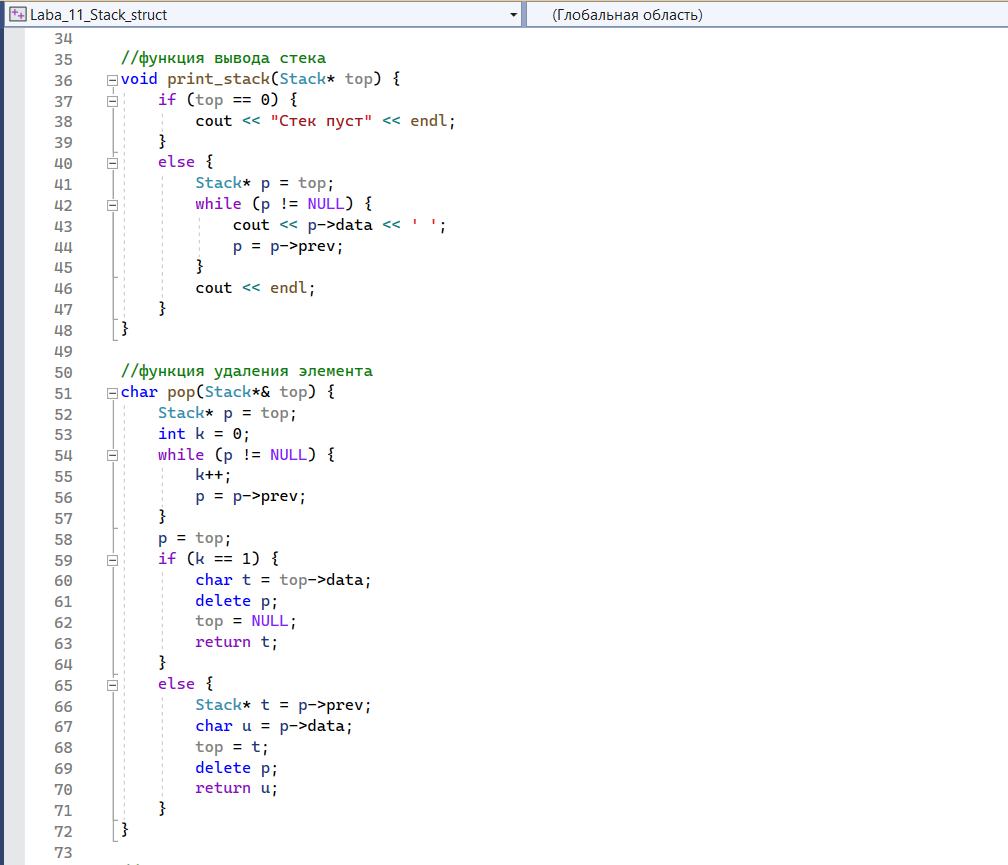
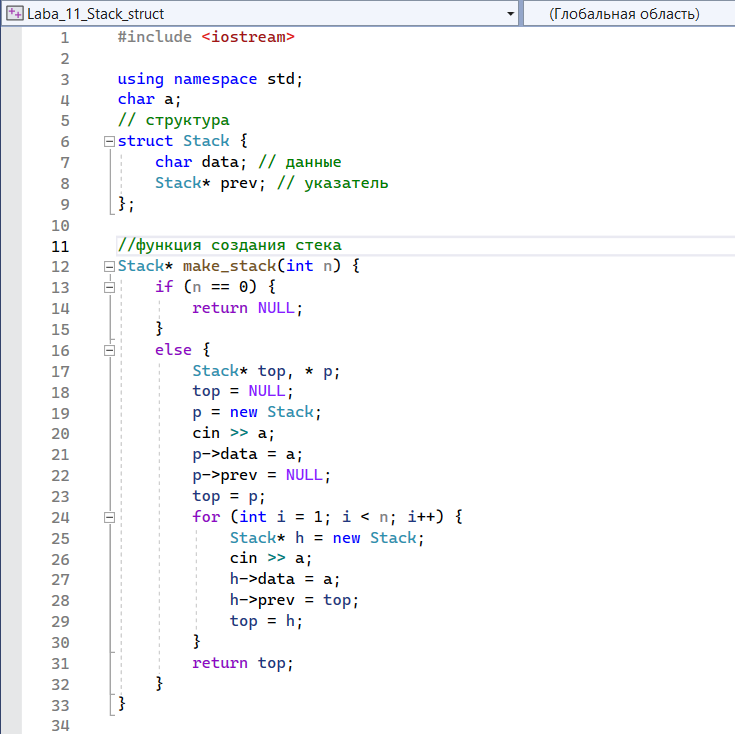




1. **Решение с помощью динамической структуры.**

Решение блок схемой:

Решение кодом:



**Решение очередью.**

Анализ задачи:

1. Создать структуру Node с полями data (данные) и next (указатель на следующий элемент).

2. Создать структуру Queue с полями size (размер), head (указатель на начало) и tail (указатель на конец).

3. Описать функцию init\_queue для инициализации очереди с одним элементом.

4. Описать функцию push для добавления элемента в конец очереди.

5. Описать функцию new\_queue для создания очереди из n элементов.

6. Описать функцию print\_queue для вывода элементов очереди.

7. Описать функцию pop для удаления элемента из начала очереди.

8. Описать функцию delete\_k для удаления всех элементов с заданным ключом.

9. Описать функцию insert для вставки K элементов перед указанным номером.

10. Описать функцию delete\_queue для освобождения динамической памяти.

11. В функции main:

11.1. Создать очередь q типа char.

11.2. Ввести количество элементов n для создания очереди.

11.3. Создать новую очередь с введенными элементами.

11.4. Вывести исходную очередь.

11.5. Ввести ключ элемента, который нужно удалить, и удалить все такие элементы.

11.6. Вывести очередь после удаления элементов.

11.7. Ввести номер элемента и количество элементов K, которые нужно вставить перед ним.

11.8. Вставить K элементов перед указанным номером.

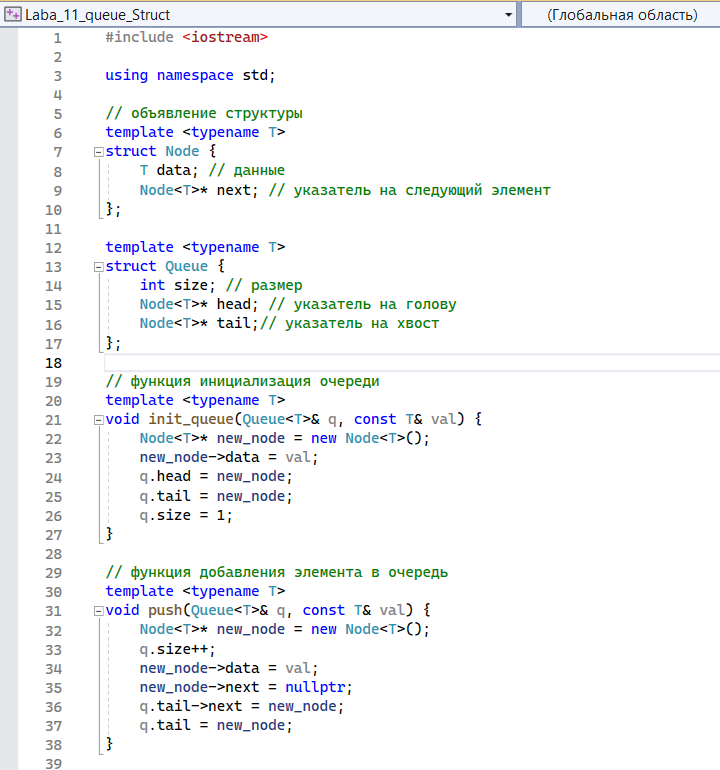
11.9. Вывести очередь после вставки элементов.

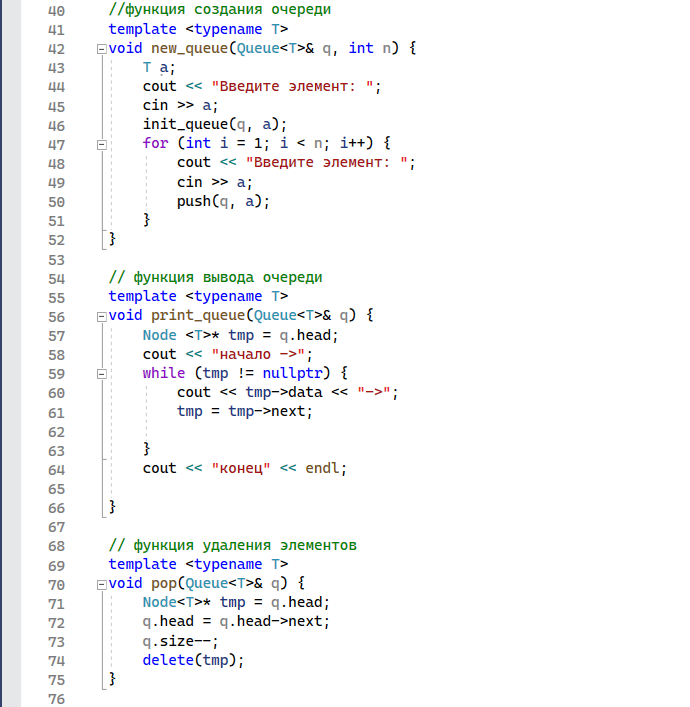
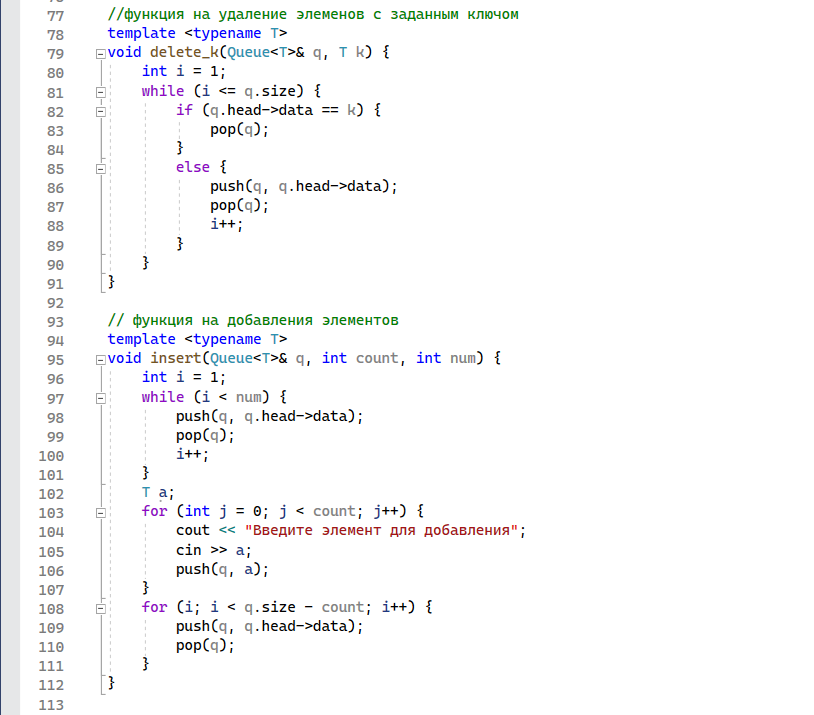
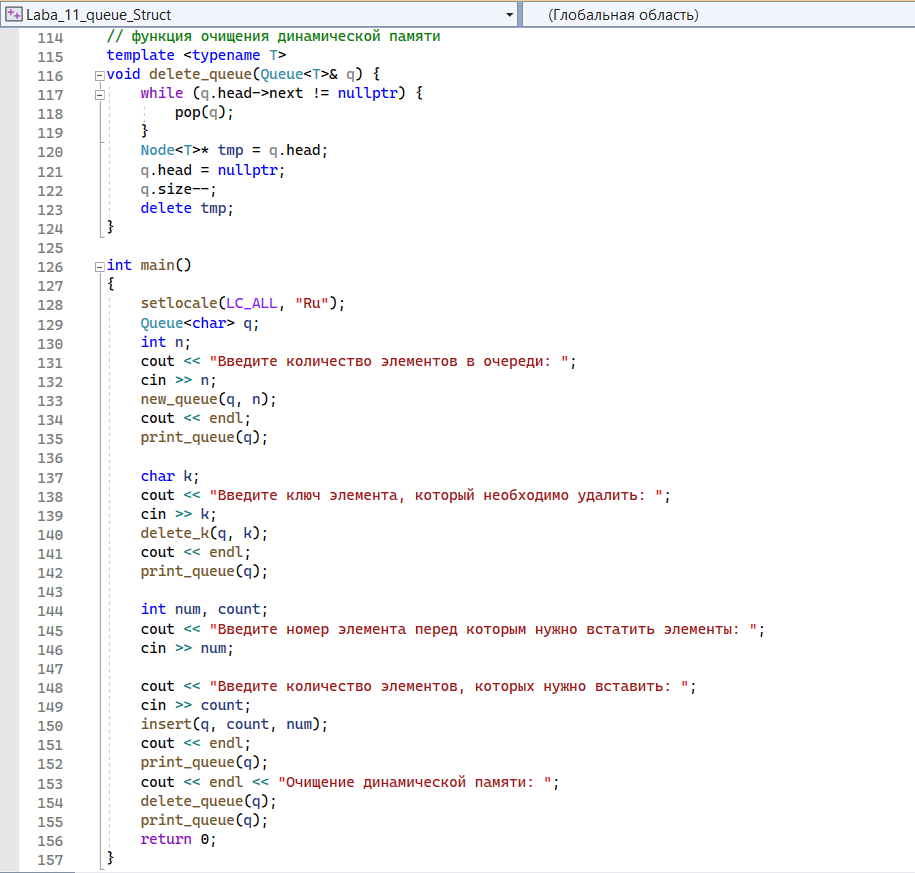
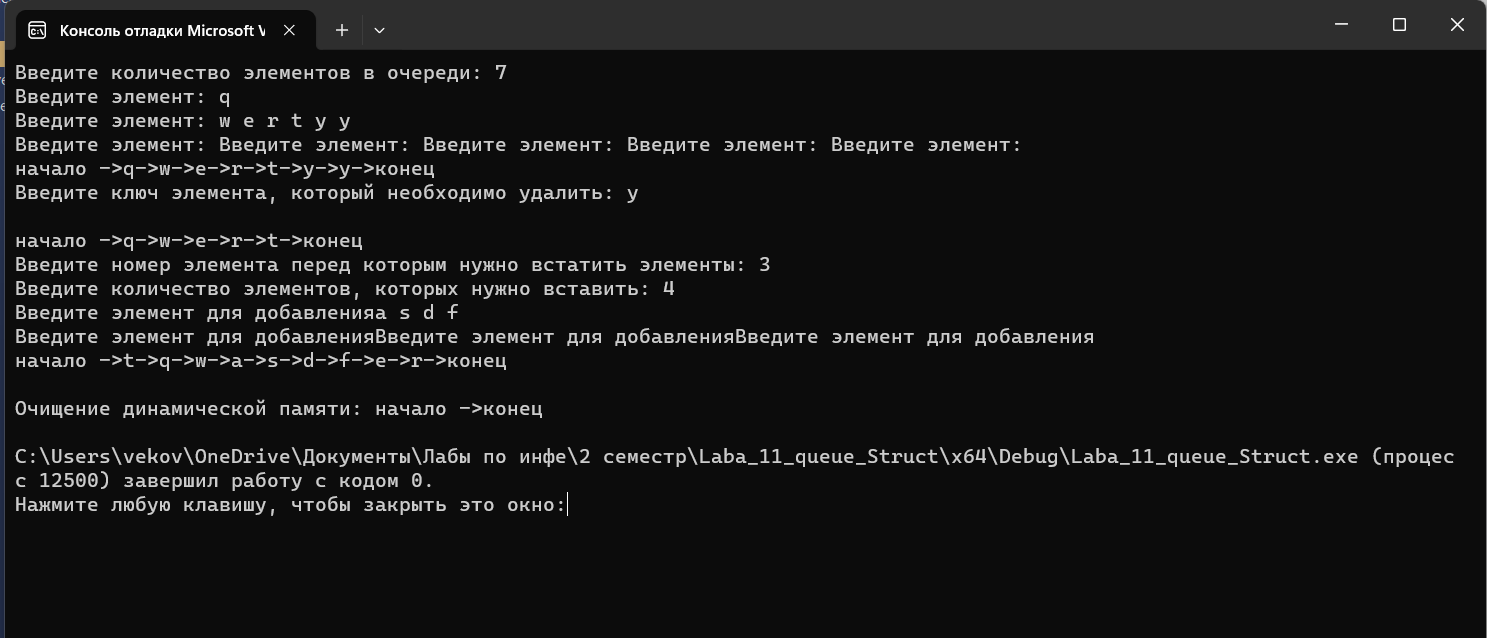
11.10. Освободить динамическую память, вывести очередь.

1. **Решение с помощью динамической структуры:**

Решение блок схемой:

Решение кодом:

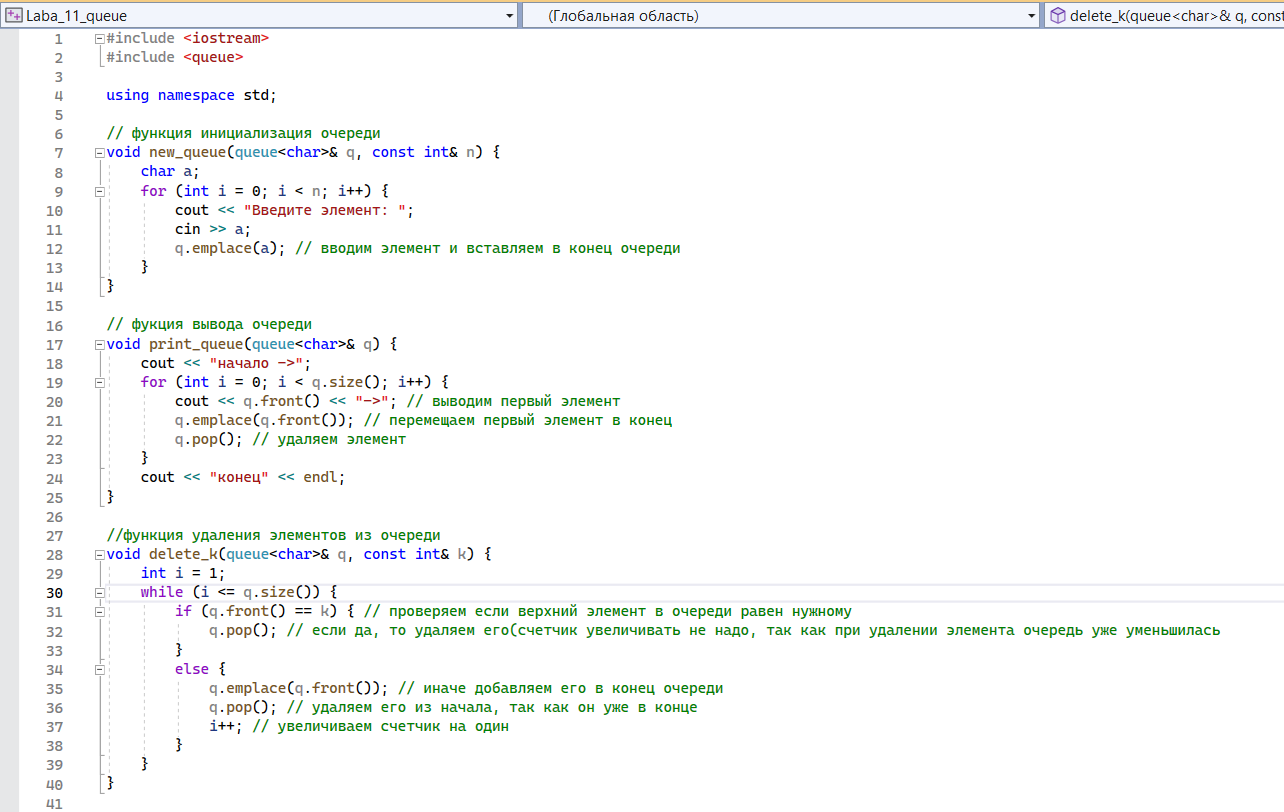


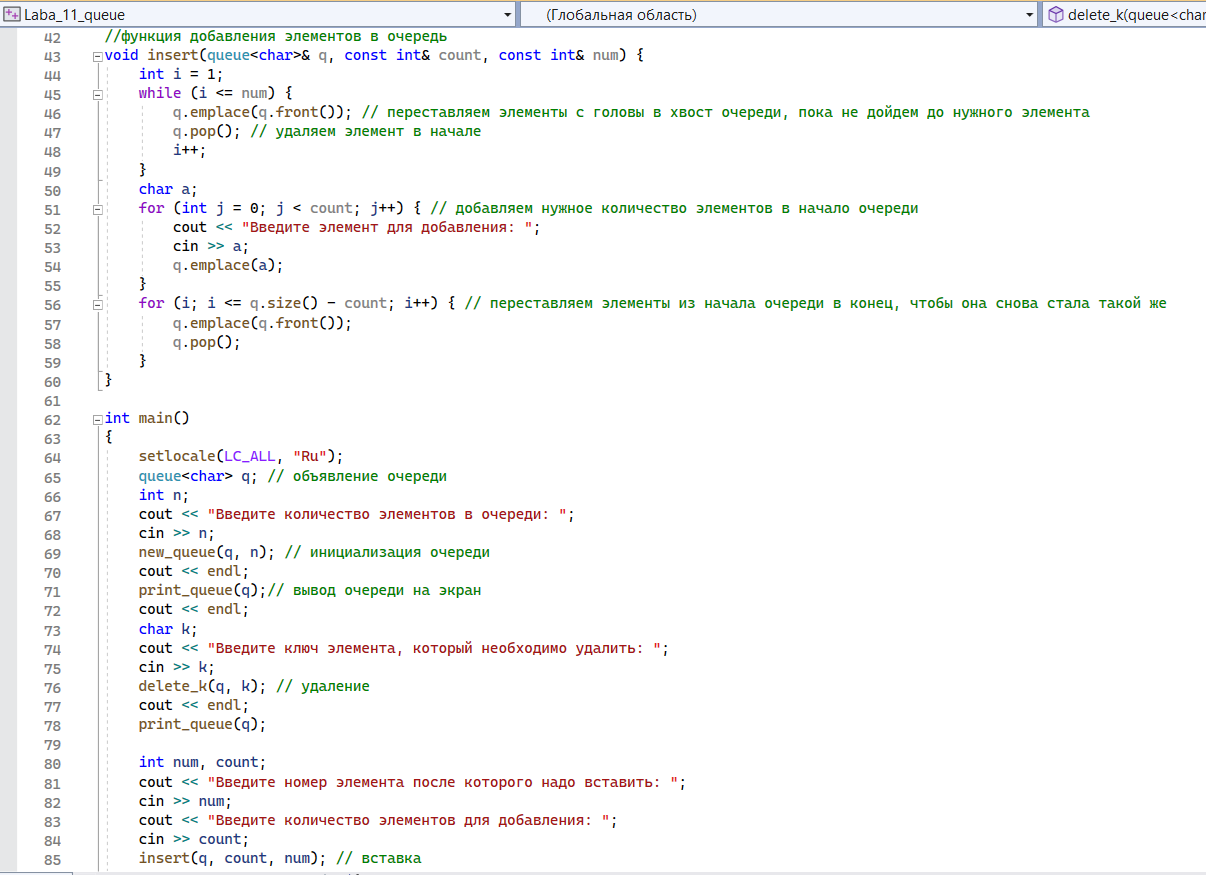
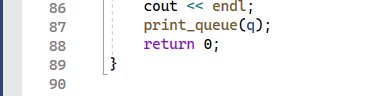
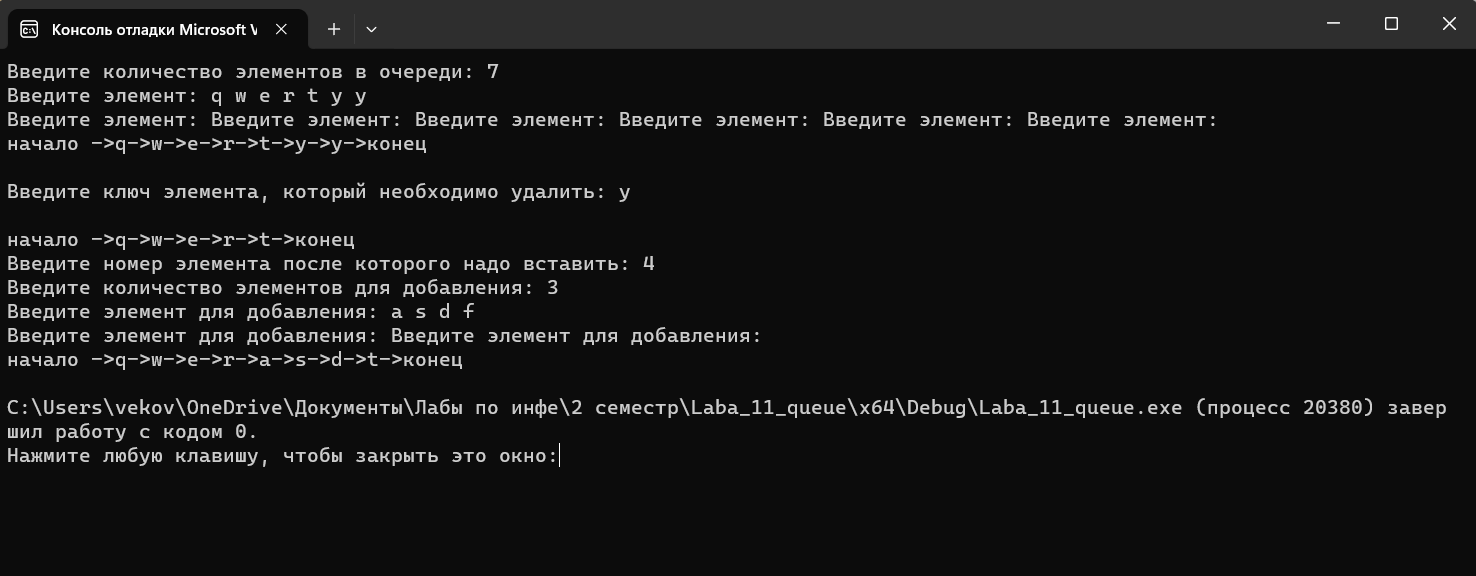
   

1. **Решение с помощью библиотеки <queue>**

Решение блок схемой:

Решение кодом:



Вывод: у меня получилось решить поставленную задачу.

Скриншот из гита: