Министерство высшего образования и науки Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

**по выполненной лабораторной работе «ОЧЕРЕДЬ»**

Дисциплина: основы алгоритмизации и программирования

Выполнил:

Студент группы ИВТ

Голубцов Никита

Проверил:

Доцент кафедры ИТАС

Полякова Ольга Андреевна

**Пермь, 2023**

**Постановка задачи**

Создать очередь.

Удалить из очереди элемент по введенному пользователем значению.

Добавить в начало очереди элемент, введенный пользователем.

**Анализ задачи**

1. Пользователь вводит количество элементов в очереди.

2. Создаем очередь с помощью функции void new\_queue().

Функция на вход получает указатель на очередь и количество в ней элементов.

Пользователь вводит первый элемент очередь. Вызывается функция void init\_queue(), которая инициализирует первый элемент очереди. Ее изначальный размер равен 1.

Затем, с помощью цикла for вводятся следующие элементы очереди, которые добавляются в очередь с помощью функции void push().

Функция void push() получает на вход указатель на очередь и значение элемента, которое необходимо добавить.

Получив новый элемент, создаем новый узел Node(). Его информационное поле – новый элемент, полученный функцией. Переобновляем значения хвостового и головного элемента.

3. Выводим очередь с помощью функции void print\_queue(). Функция получает на вход указатель на очередь. С помощью цикла while с условием неокончания очереди, выводятся значения каждого узла.

4. Вызываем функцию void delete\_key()которая ищет элемент с счетным заданным полем и удаляет его.

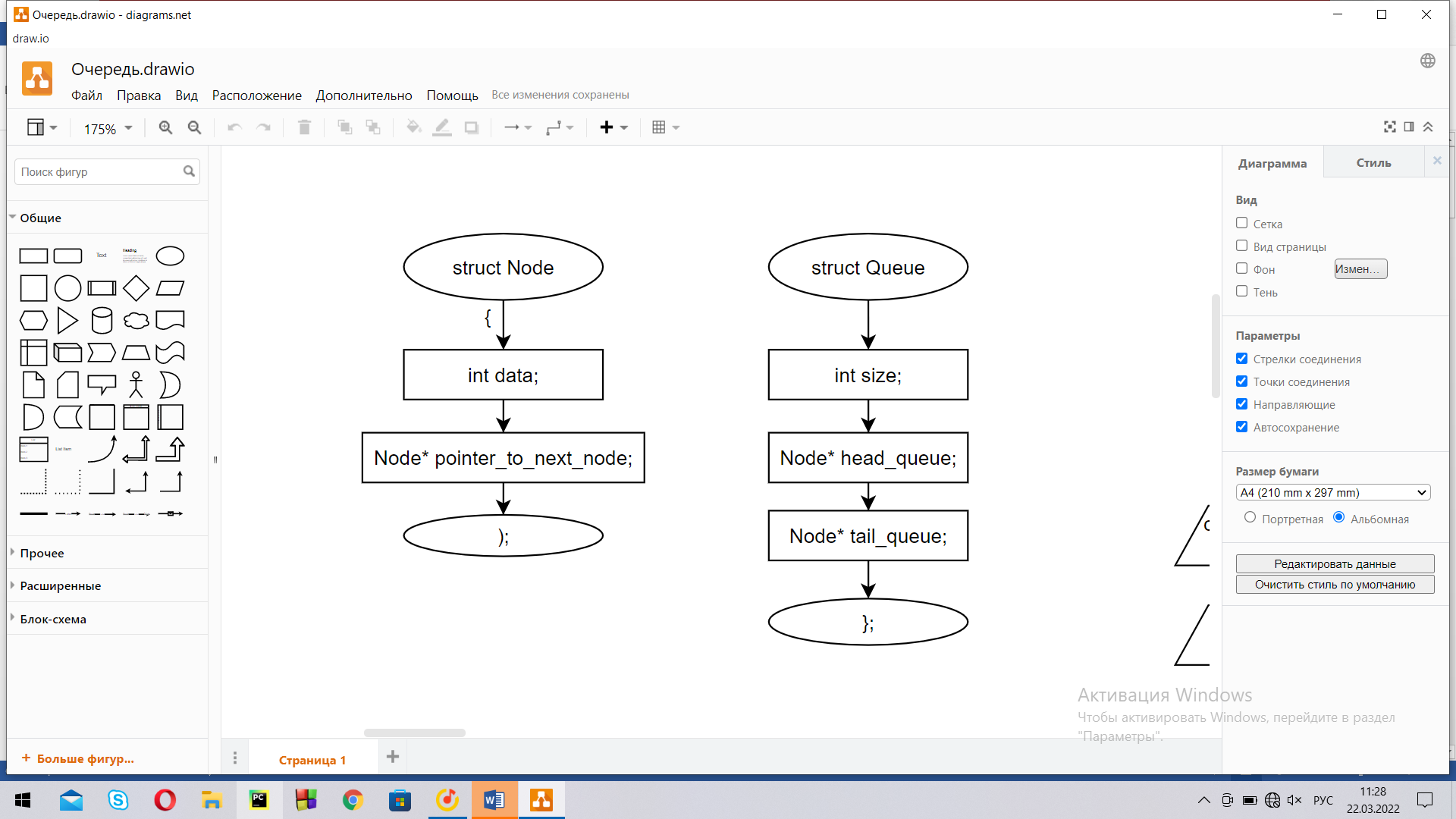
Если первый элемент равен ключу, он удаляется. Иначе - элемент в конец очереди

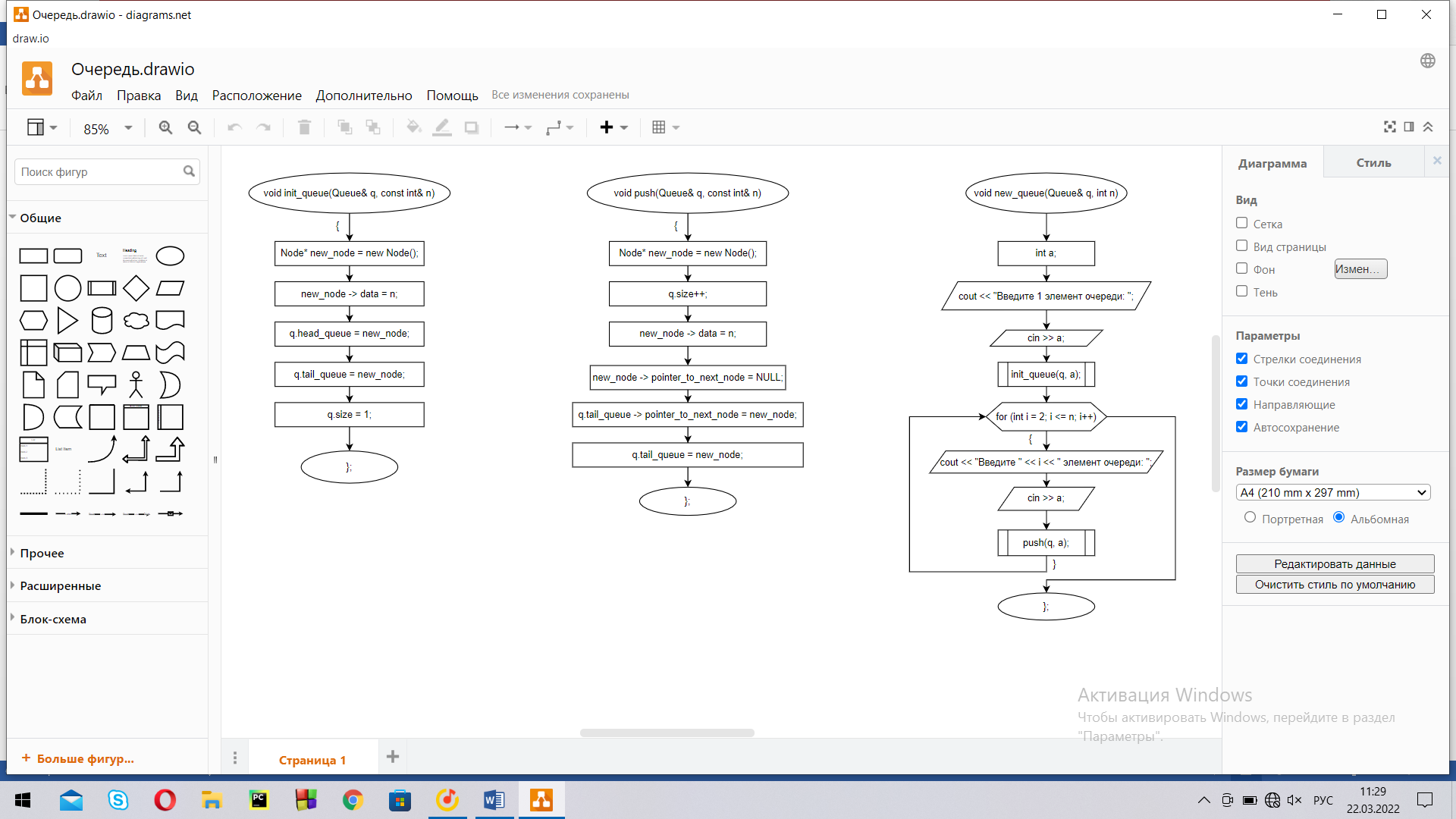
Проходя по всем элементам очереди, функция проверяет, является ли информационное поле данного элемента.

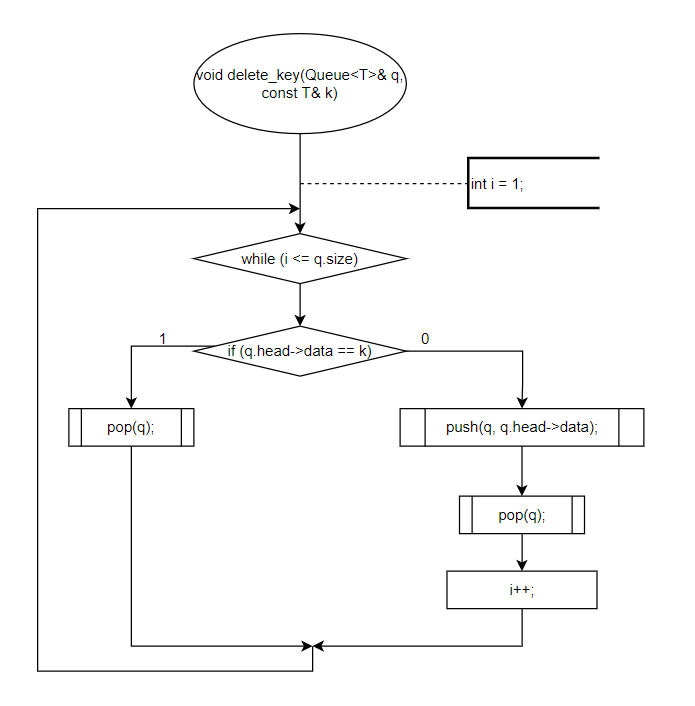
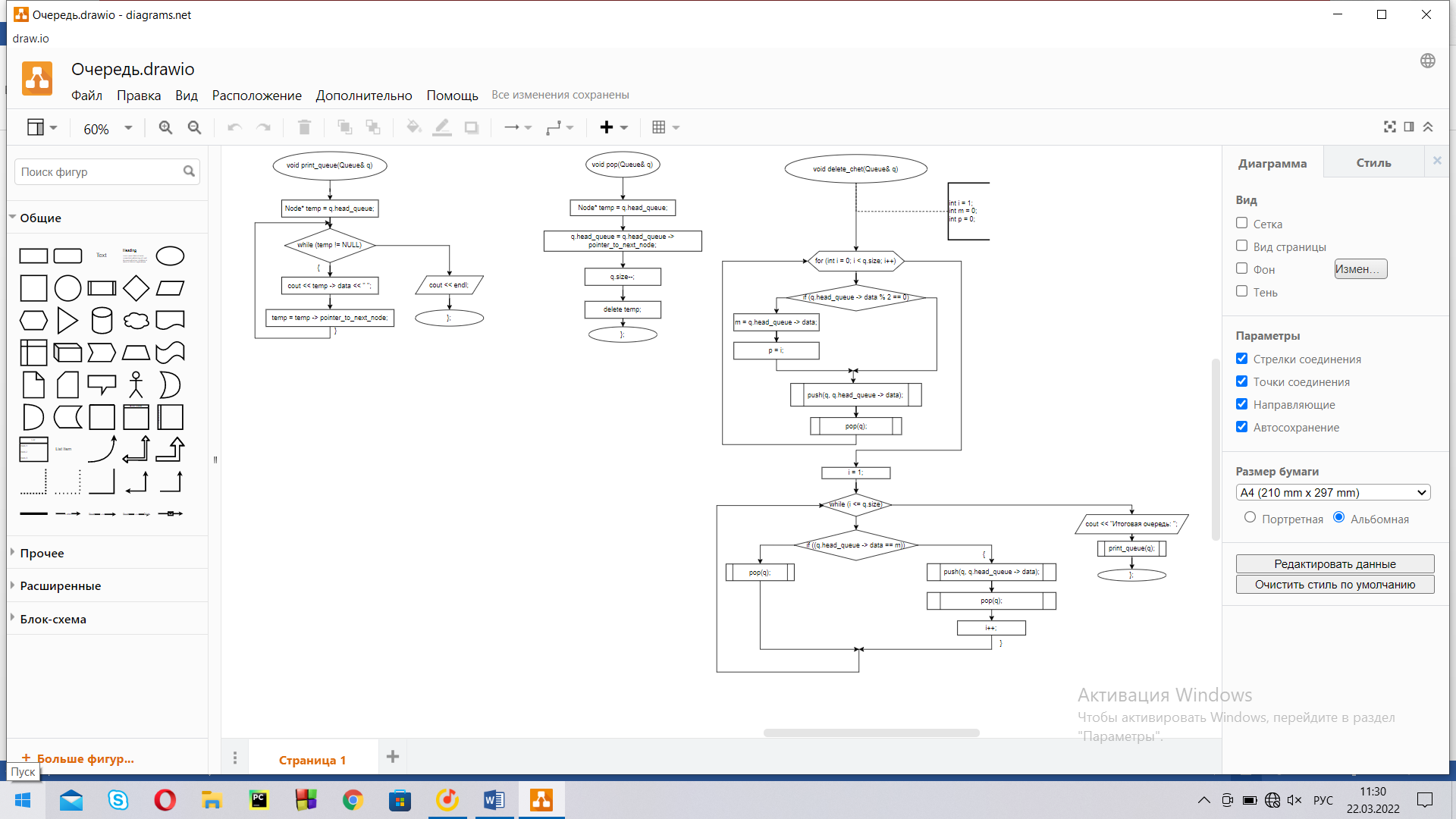
5. С помощью цикла while проходим по очереди повторно, проверяя, является ли данный элемент очереди запомненным ранее. Если он является таковым, то удаляем его из очереди с помощью функции void pop().

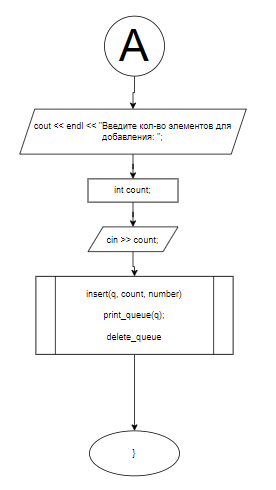
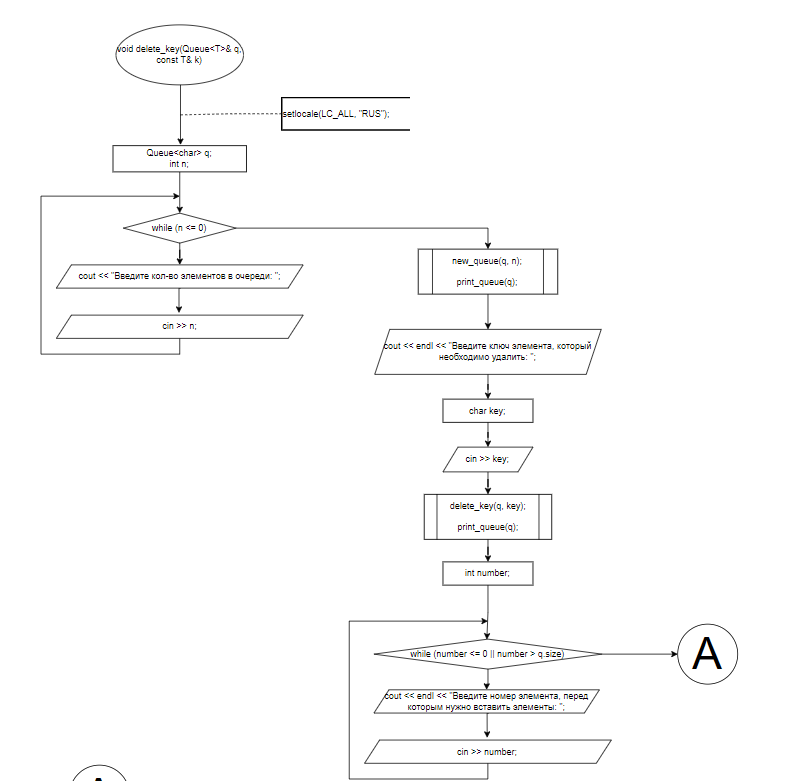
6. Добавление новых элементов осуществляется с помощью метода void insert().

**Блок – схема**

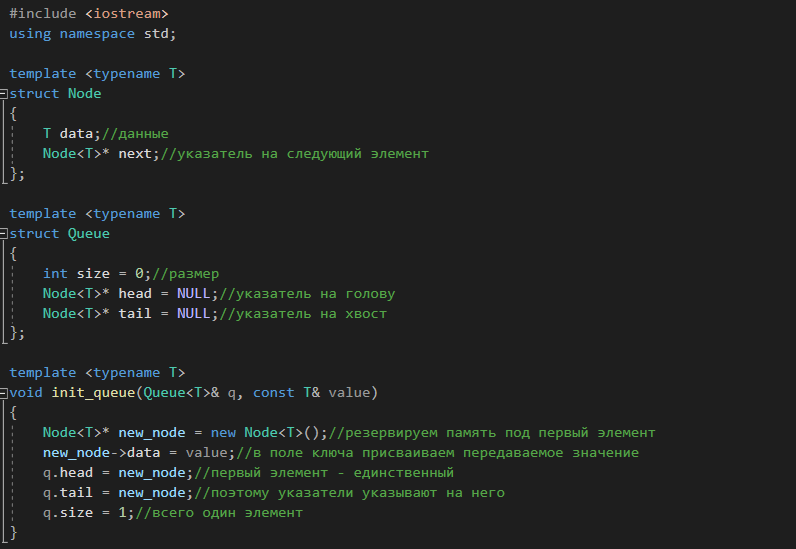


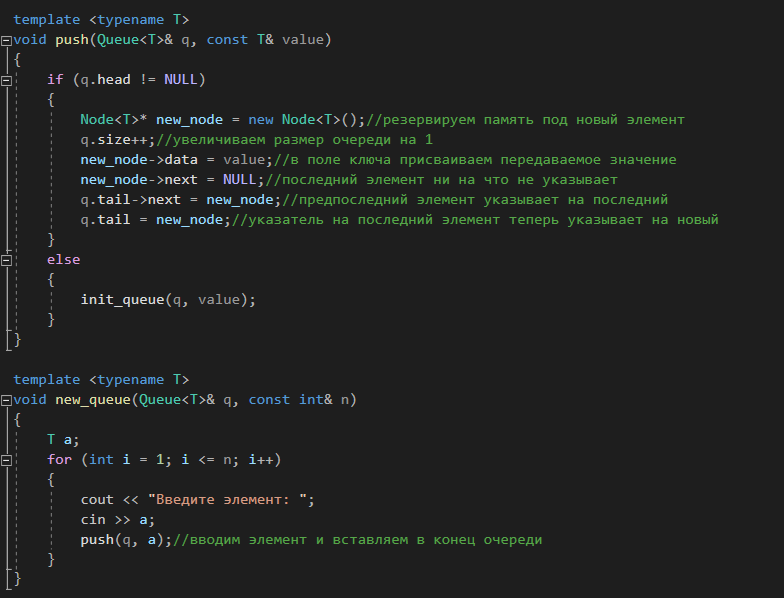


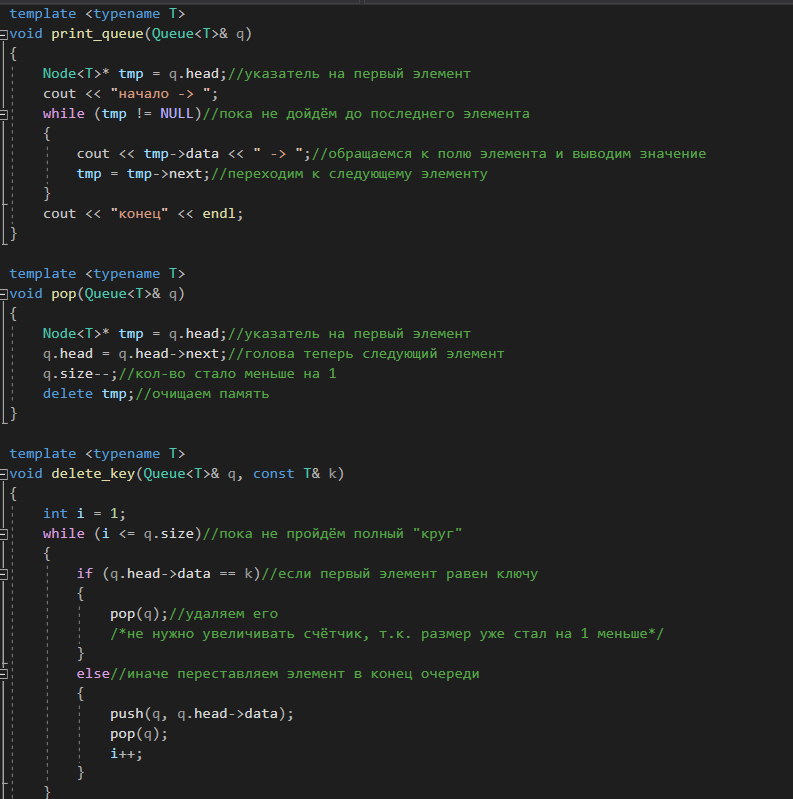


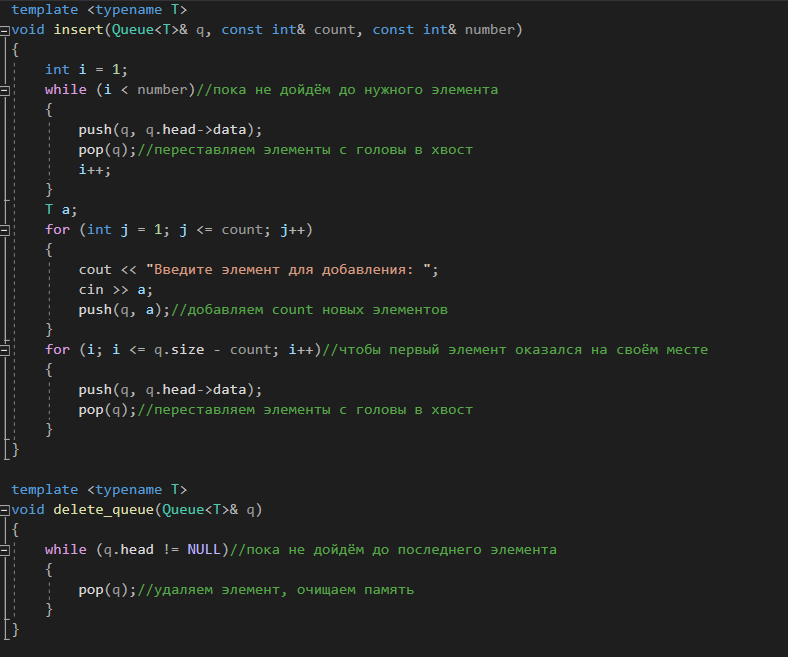


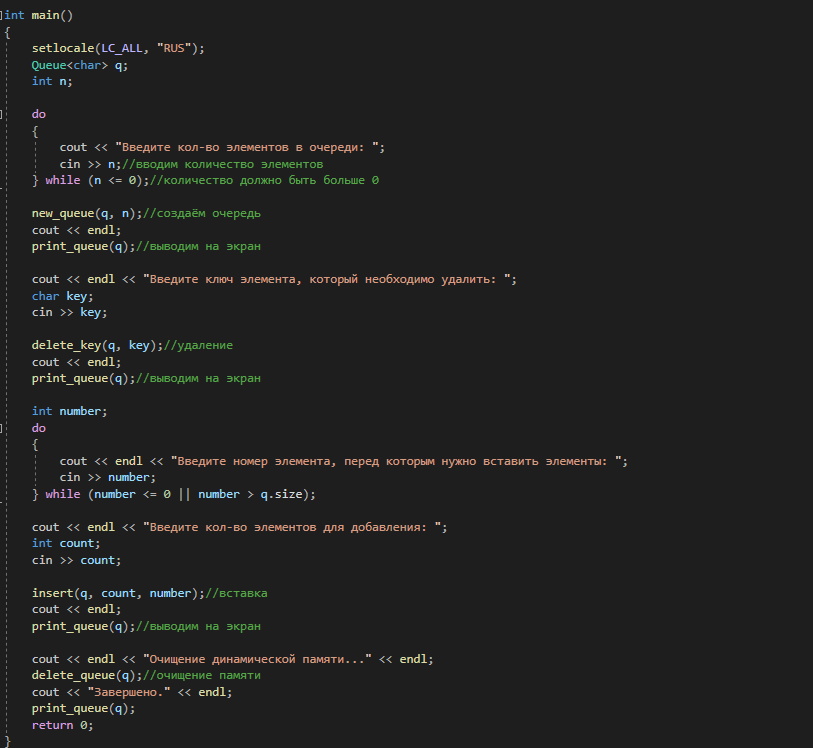
**Код программы**



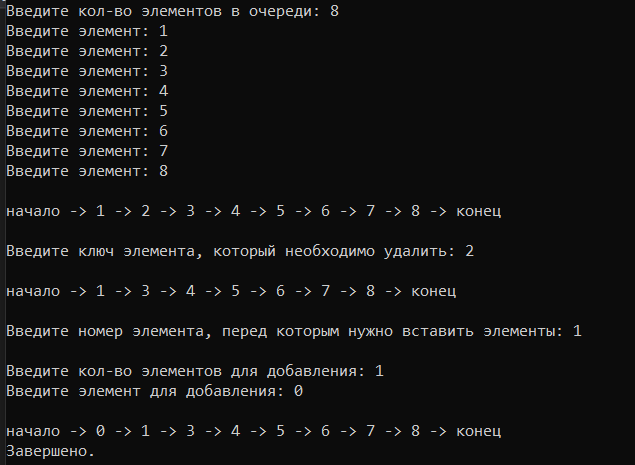








**Результаты работы программы**



**Анализ полученных результатов**

На вход получена очередь.

Пользователем выбрано удаление элемента со значением 2. Удалено первое вхождение элемента.

Далее, пользователю предлагается в любое место вставить любое количество элементов. На первое место очереди вставляется один элемент.