Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Пермский национальный исследовательский политехнический университет» (ПНИПУ)

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

**Лабораторная работа № 11.4 (две реализации)**

**по дисциплине**

**«Основы алгоритмизации и программирования»**

**семестр** 2

Выполнил студент гр. ИВТ-21-1б

Балдин Алексей Игоревич

Проверил:

Яруллин Денис Владимирович

\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

(оценка) (подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

г. Пермь - 2022

**Цель**

Получить практические навыки работы с очередями. Очередь – это частный случай однонаправленного списка.

**Постановка задачи**

Сформировать очередь. Тип информационного поля указан в варианте. Распечатать полученную структуру. Выполнить обработку структуры в соответствии с заданием. Распечатать полученный результат.

**Исходные данные**

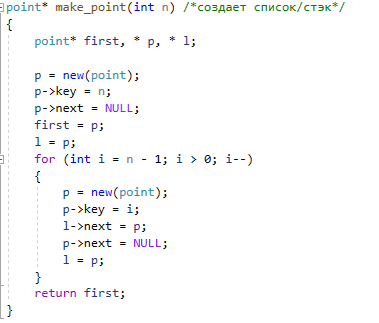
Вариант 5:

Сначала представлю реализацию и блок-схему к очереди (БЕЗ библиотеки), после, будет очередь (С ПОМОЩЬЮ библиотеки).

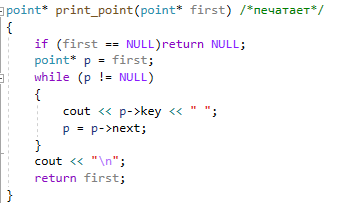
Тип информационного поля int. Удалить из очередь все элементы с четными информационными полями.

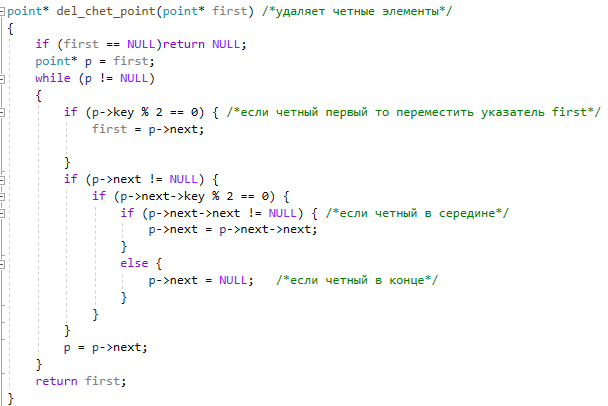
**Итак, реализация очереди (без библиотеки):**

Данная функция создает очередь, то есть создается новый элемент очереди, в него забрасывается информационное поле, указатель на следующий элемент равен 0. Так как в моей реализации создаю первый элемент очереди до цикла, то в цикле for иду от (n-1) элемента.

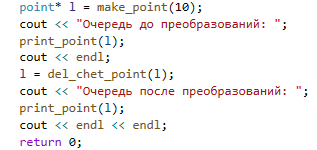


Функция выводит введенную очередь на экран, при условии, что указатель на конец очереди не нуль, иначе возвращаем нуль. В цикле не забываем переместить указатель на следующий. Функция возвращает указатель в конец очереди.

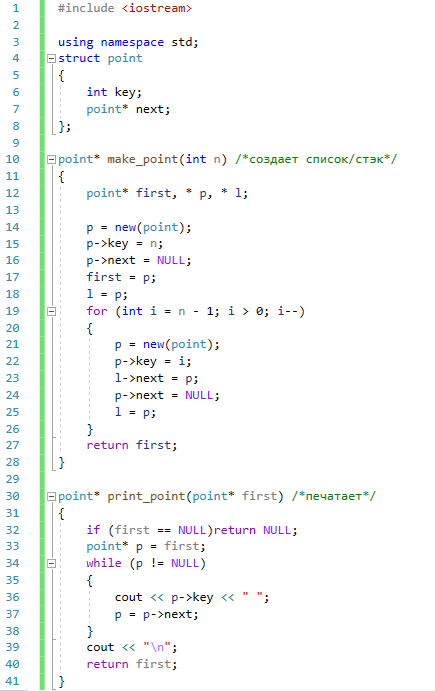


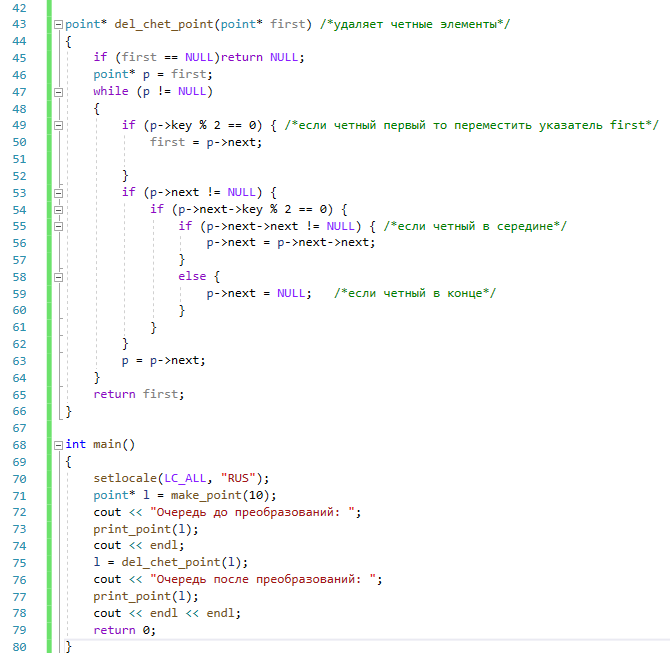
Данная функция удаляет элементы четные элементы. Создаем вспомогательный указатель (p) – дубликат указателя на конец очереди. В цикле пока указатель на конец не равен нулю, переходим (if), и проверяем элемент очереди на чётность, если четный, то указатель перемещаем на следующий, как бы пропуская четные элементы. Также есть проверка на четность в середине и в конце.

В (main) через дополнительную структуру обращаюсь ко всем функциям, и вывожу результат на экран.

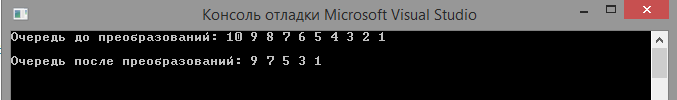


Код программы:





Результат работы программы:



Результат 1

Блок-схема к программе:

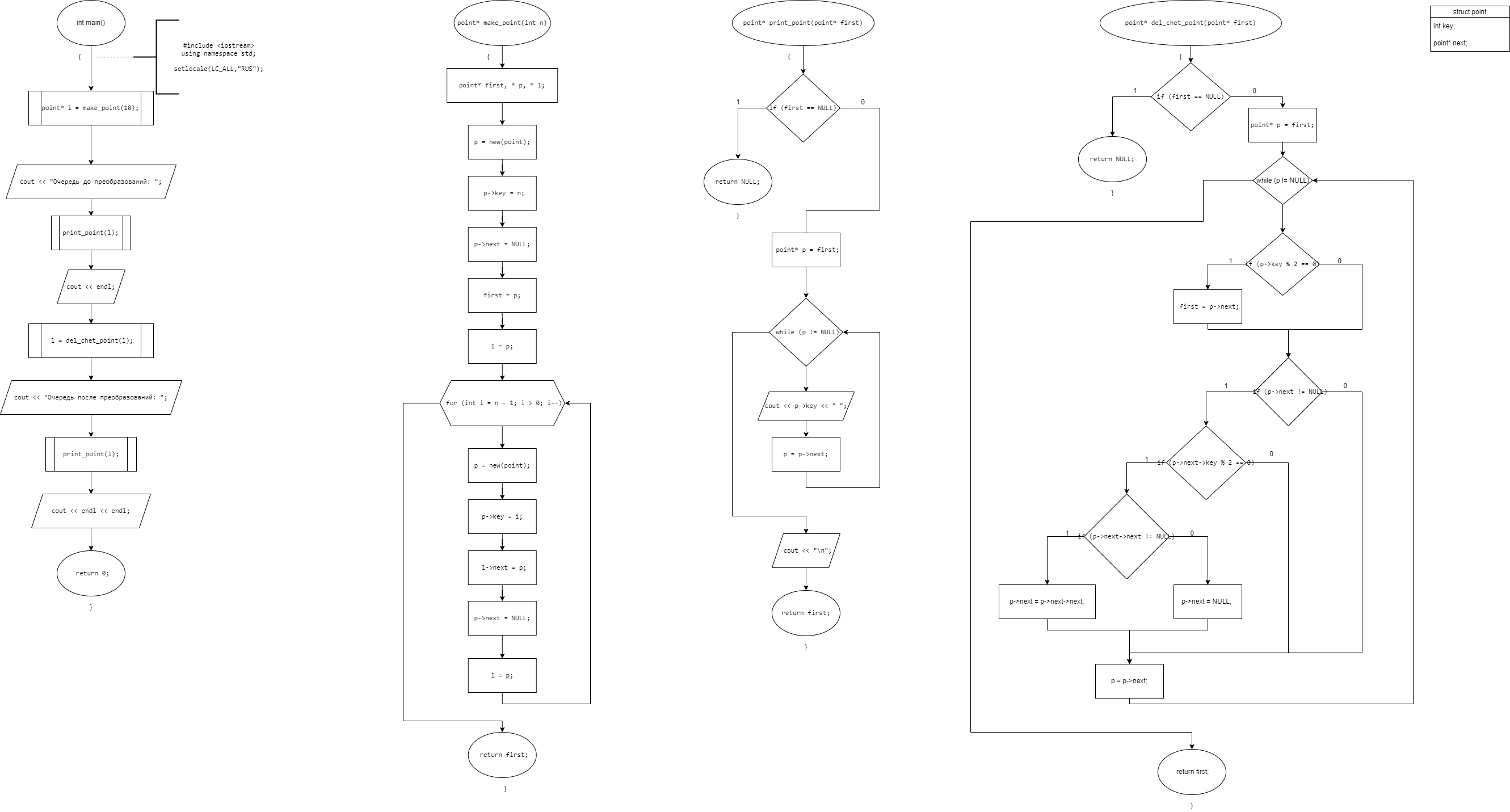


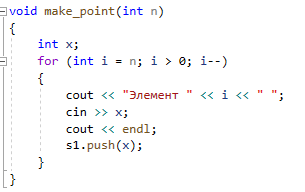
Рисунок 1

**Итак, реализация очереди (С помощью библиотеки):**

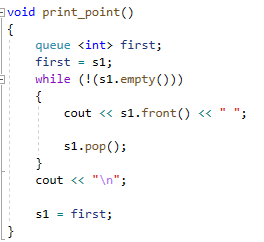
Когда мы работаем с библиотекой, очередь создаётся таким образом:



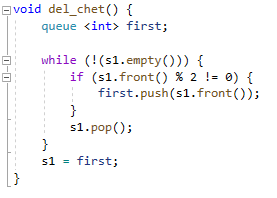
Здесь функция создает очередь, как и в прошлой реализации, просто теперь мы пользуемся указателями не в ручную, а с помощью встроенных функций библиотеки «queue», а именно s1.push – значит заполняем очередь элементами, которые мы ввели.



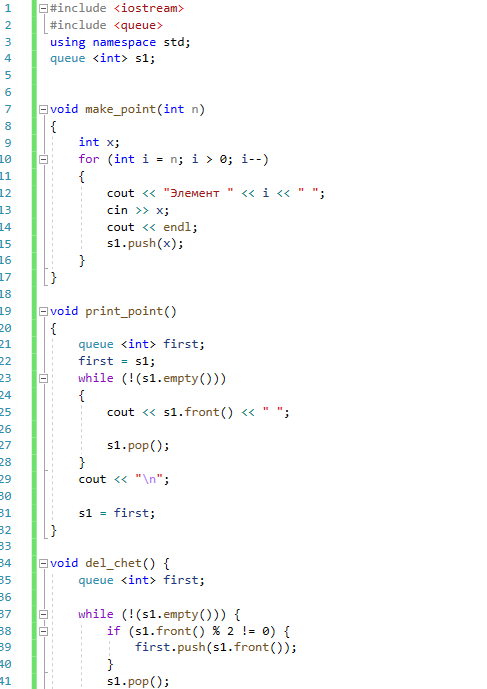
Данная функция выводит очередь на экран, при условии, что конец очереди не пуст, после вывода на экран последнего (по идее последний, но вводим мы его первым) элемента, мы его удаляем, с помощью функции (pop), тем самым наш элемент становится на элемент ниже, и так происходит в цикле, в результате выводим элементы очереди на экран.

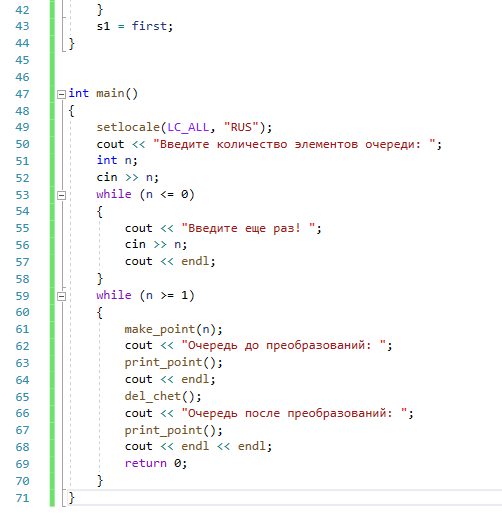


Здесь функция удаляет чётные элементы, то есть начиная с нашего первого введенного элемента, проверяем на нечетность его и последующие элементы, а чтобы проверить следующие элементы, мы удаляем уже проверенный первый элемент, и так проверяем следующие. То есть если элемент нечетный, то мы его оставляем. После прохода цикла возвращаем наш первый элемент. Как раз (front) – позволяет нам забрасывать элементы в начало очереди.

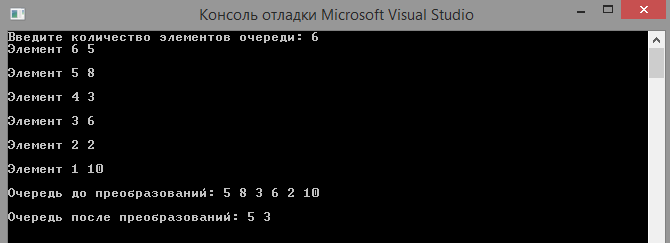


Код программы:





Результат работы программы:



Результат 2

Блок-схема к задаче:

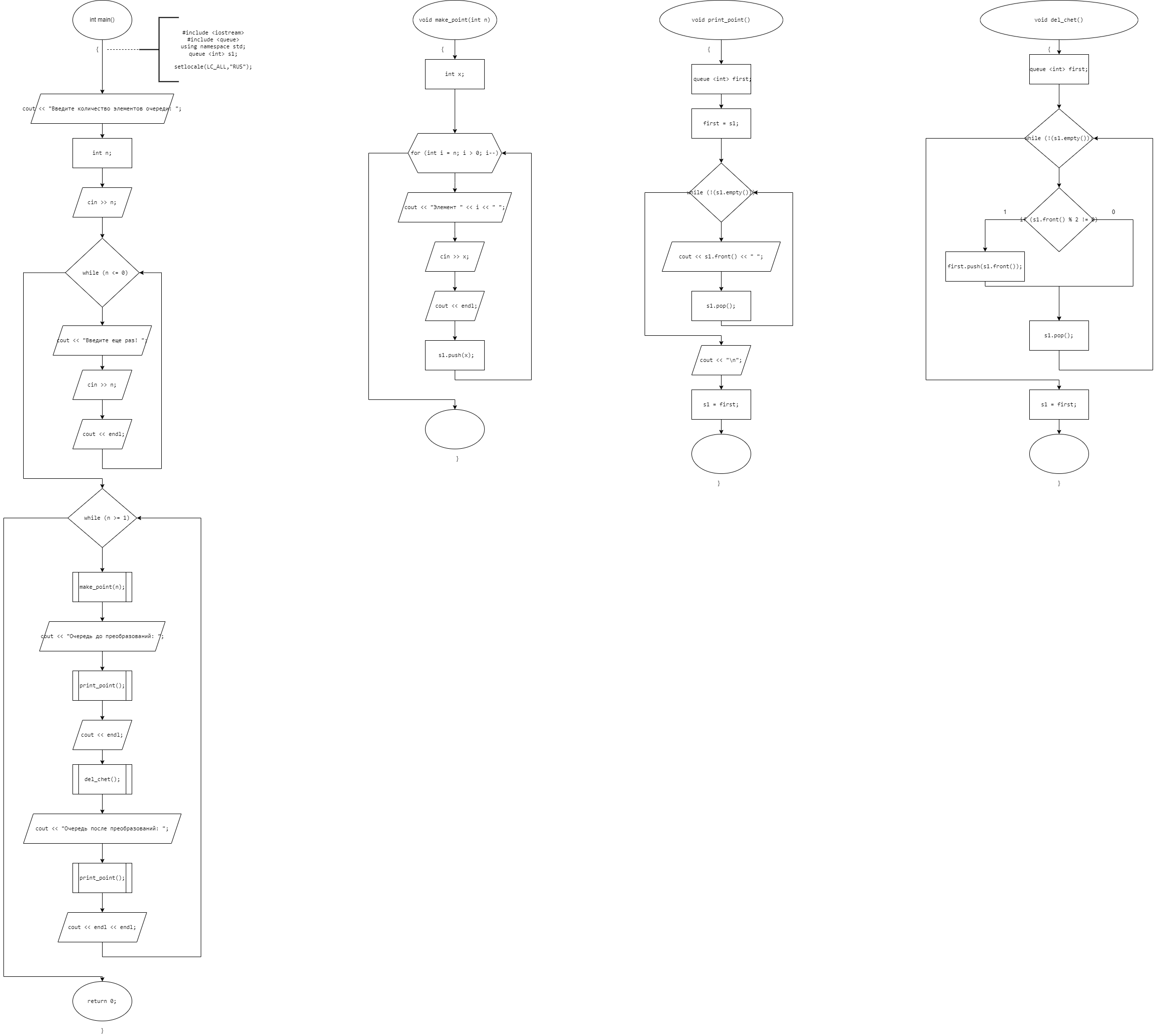


Рисунок 2