

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА – Российский технологический университет»**

**РТУ МИРЭА**

**Отчет по выполнению практического задания номер один**

**Тема: Внешние сортировки**

**Дисциплина: Структуры и алгоритмы обработки данных**

Выполнил студент Михайлюк Д. С.

Группы ИНБО-07-21

Москва 2021

**СОДЕРЖАНИЕ**

[1 ПОСТАНОВКА ЗАДАЧИ 3](#_Toc100793787)

[2 Определение списка операций над списком, которые выявлены в процессе исследования задач дополнительного задания 4](#_Toc100793788)

[2.1 Проверка списка на пустоту 4](#_Toc100793789)

[2.2 Метод добавления узла в начало 5](#_Toc100793790)

[2.3 Метод добавления узла в конец 7](#_Toc100793791)

[2.4 Метод вывода списка на экран 9](#_Toc100793792)

[2.5 Метод поиска узлов по значению 11](#_Toc100793793)

[2.6 Метод удаления первого узла 12](#_Toc100793794)

[2.7 Метод удаления последнего узла 14](#_Toc100793795)

[2.8 Метод удаления узла по его значению 16](#_Toc100793796)

[2.9 Метод перемещения первых n-узлов в конец списка 19](#_Toc100793797)

[2.10 Метод удаления предпоследнего узла 21](#_Toc100793798)

[2.11 Метод обмена местами 2х узлов 23](#_Toc100793799)

[2.12 Метод обмена местами узлов с наибольшим и наименьшим значениями 25](#_Toc100793800)

[3 КОД ВСЕЙ ПРОГРАММЫ 27](#_Toc100793801)

[4 ВЫВОДЫ 32](#_Toc100793802)

[5 ОТВЕТЫ НА ВОПРОСЫ 33](#_Toc100793803)

# ПОСТАНОВКА ЗАДАЧИ

Реализовать программу решения задачи по использованию линейного однонаправленного списка на языке программирования C++. Включить в код программы обязательный фрагмент, определенный требованием в задаче и дополнительные функции, назначенные согласно персональному варианту.

Персональный вариант - №8:

Дан линейный однонаправленный список L1.

Разработать функцию, которая переносит первые k узлов в конец списка.

Разработать функцию, которая переставляет местами узлы с максимальным и минимальным значениями.

Разработать функцию, которая удаляет предпоследний узел списка.

# Определение списка операций над списком, которые выявлены в процессе исследования задач дополнительного задания

## Проверка списка на пустоту

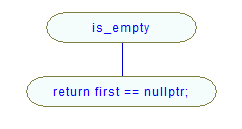


Рисунок 1 – Блок схема для метода

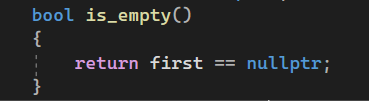


Рисунок 2 – Код метода

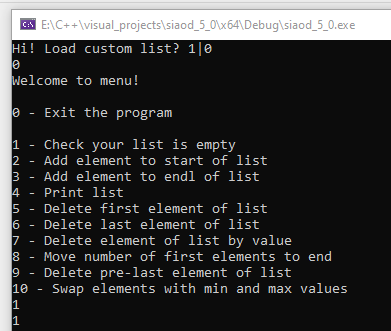


Рисунок 3 – Тест метода

## Метод добавления узла в начало

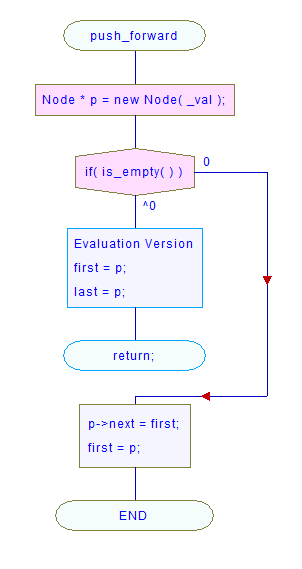


Рисунок 4 – Блок схема для метода

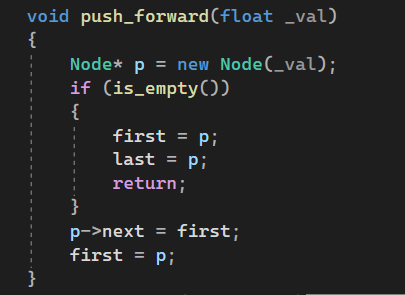


Рисунок 5 – Код метода

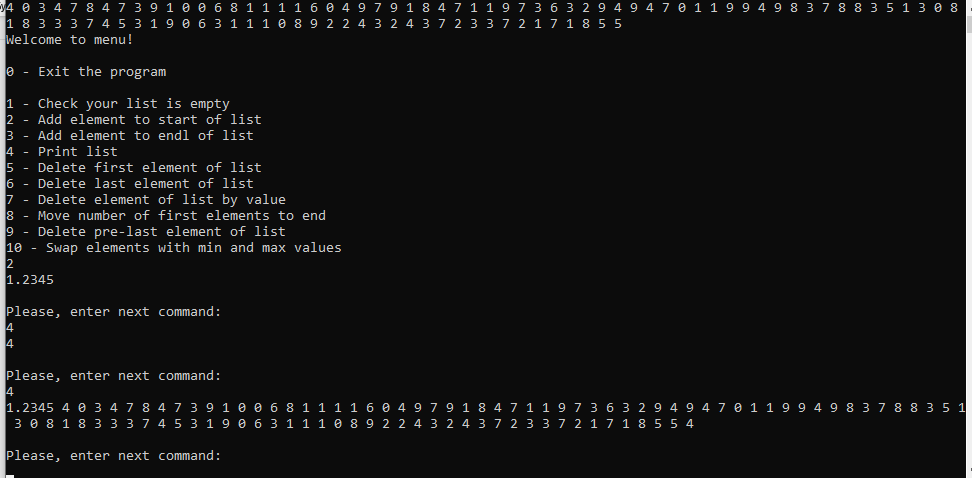


Рисунок 6 – Тест метода

## Метод добавления узла в конец

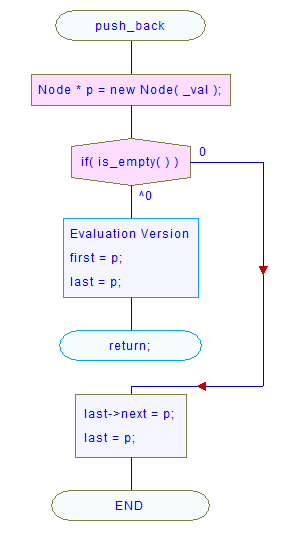


Рисунок 7 – Блок схема для метода

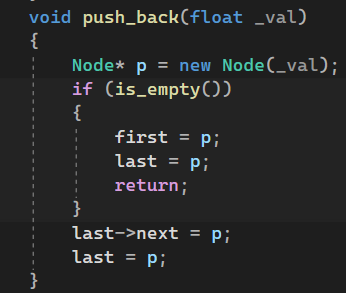


Рисунок 8 – Код метода

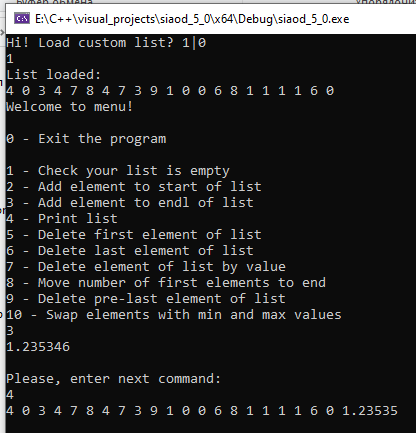


Рисунок 9 – Тест метода

## Метод вывода списка на экран

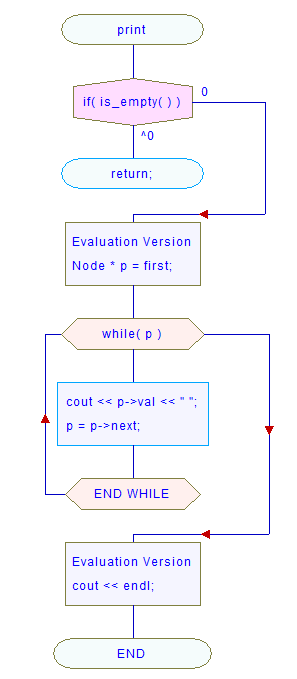


Рисунок 10 – Блок схема для метода

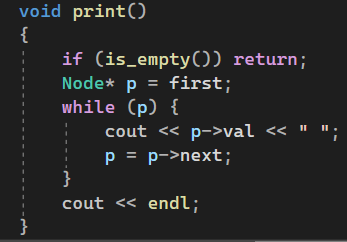


Рисунок 11 – Код метода

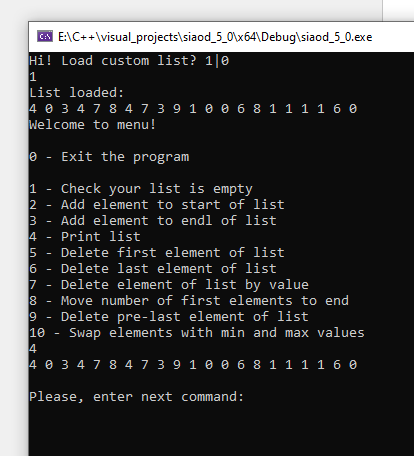


Рисунок 12 – Тест метода

## Метод поиска узлов по значению

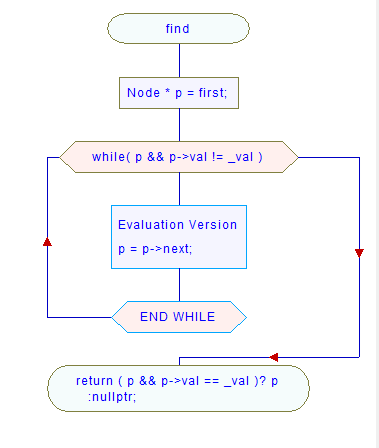


Рисунок 13 – Блок схема для метода

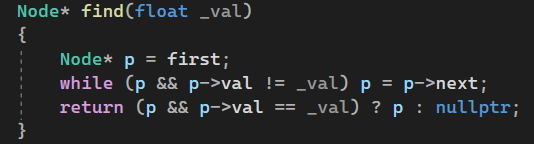


Рисунок 11 – Код метода

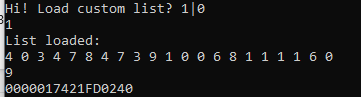


Рисунок 12 – Тест метода

## Метод удаления первого узла

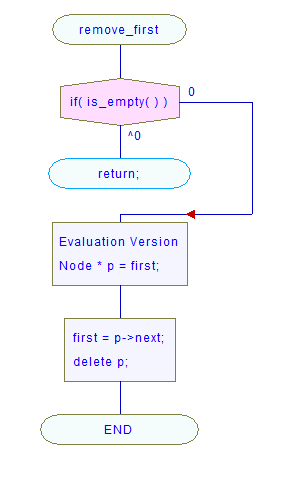


Рисунок 13 – Блок схема для метода

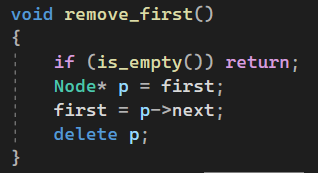


Рисунок 14 – Код метода

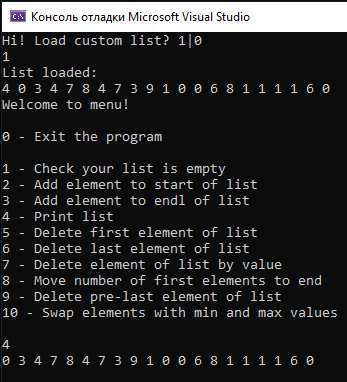


Рисунок 15 – Тест метода

## Метод удаления последнего узла

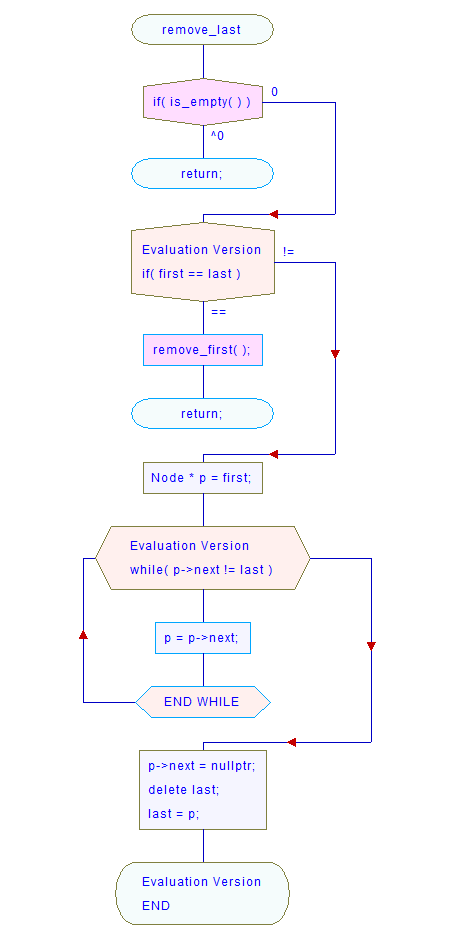


Рисунок 16 – Блок схема для метода

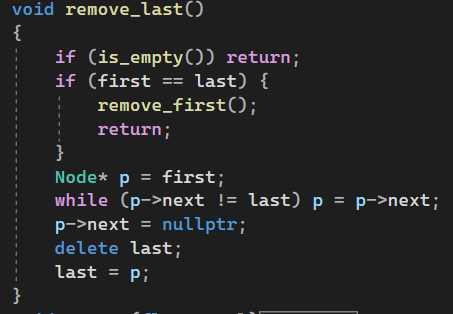


Рисунок 17 – Код метода

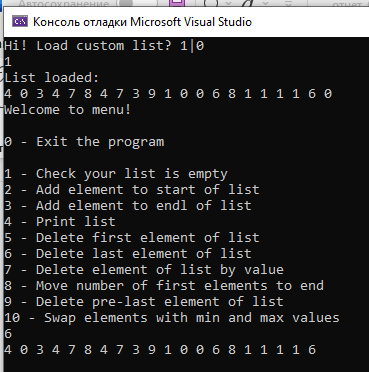


Рисунок 18 – Тест метода

## Метод удаления узла по его значению

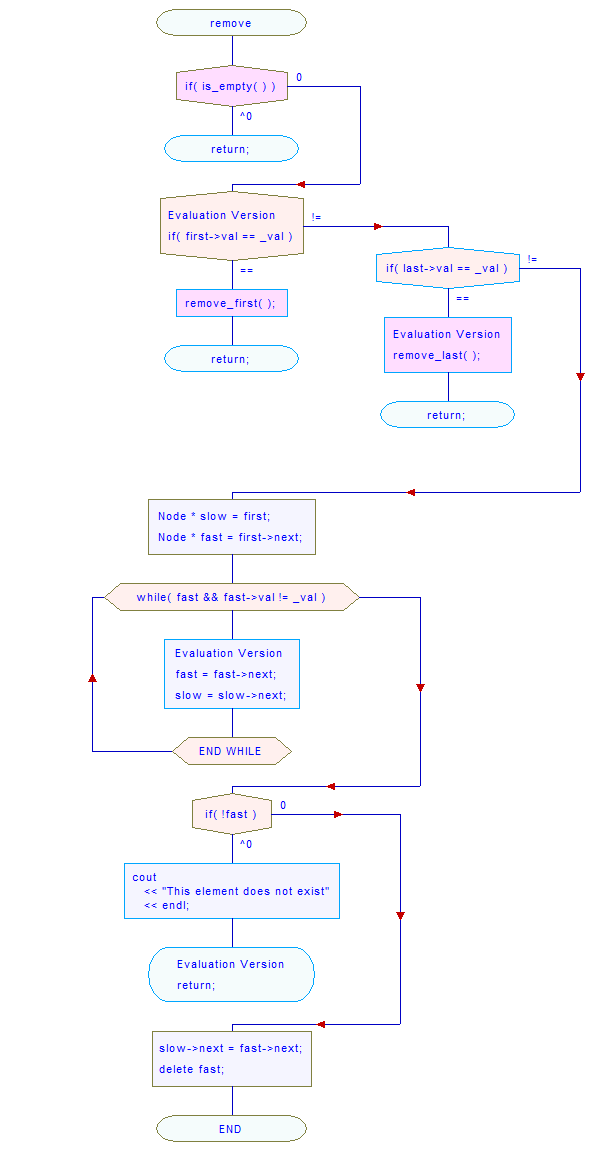


Рисунок 19 – Блок схема для метода

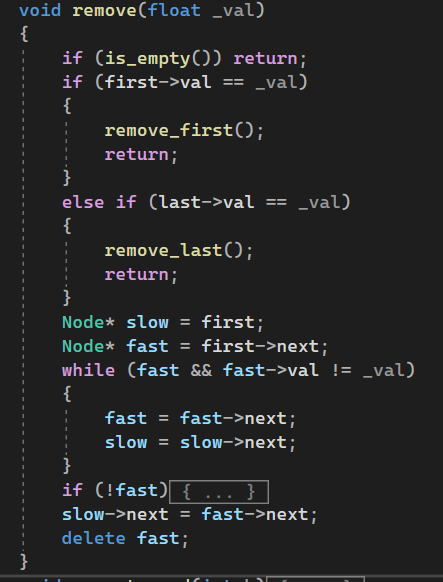


Рисунок 20 – Код метода

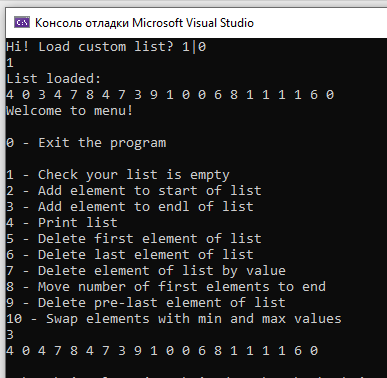


Рисунок 21 – Тест метода

## Метод перемещения первых n-узлов в конец списка

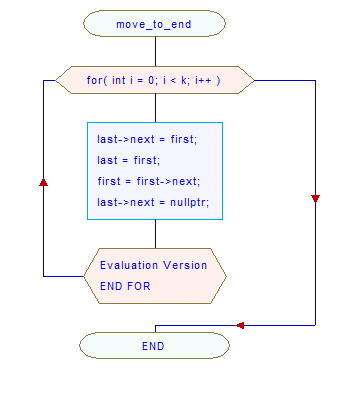


Рисунок 22 – Блок схема для метода

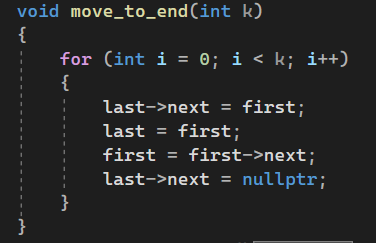


Рисунок 23 – Код метода

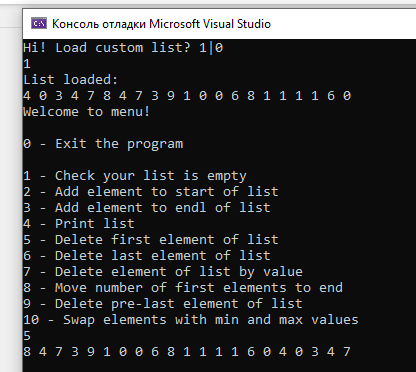


Рисунок 24 – Тест метода

## Метод удаления предпоследнего узла

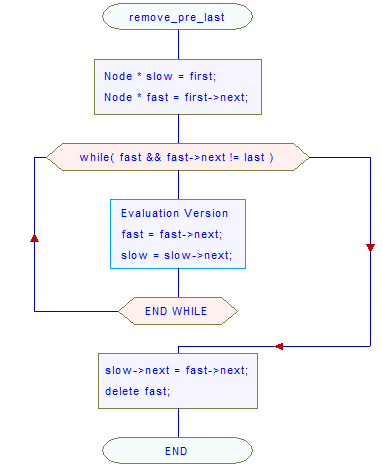


Рисунок 25 – Блок схема для метода

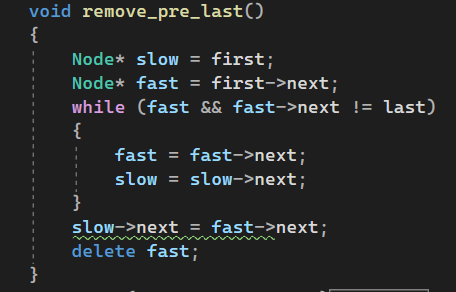


Рисунок 26 – Код метода

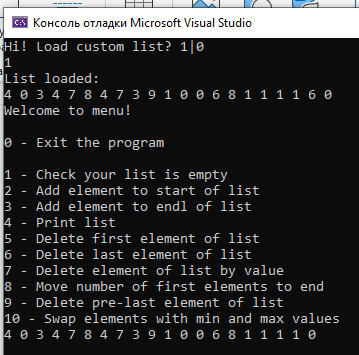


Рисунок 27 – Тест метода

## Метод обмена местами 2х узлов

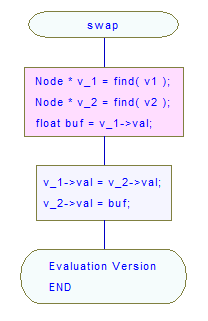


Рисунок 28 – Блок схема для метода

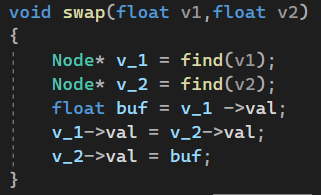


Рисунок 29 – Код метода

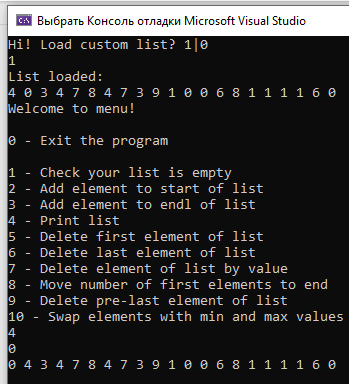


Рисунок 30 – Тест метода

## Метод обмена местами узлов с наибольшим и наименьшим значениями

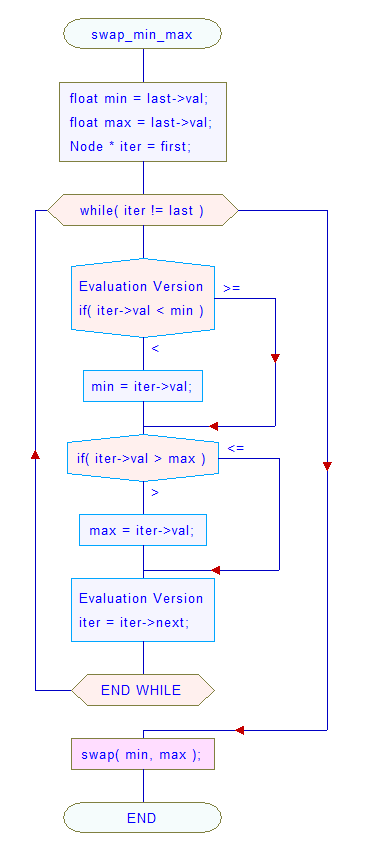


Рисунок 31 – Блок схема для метода

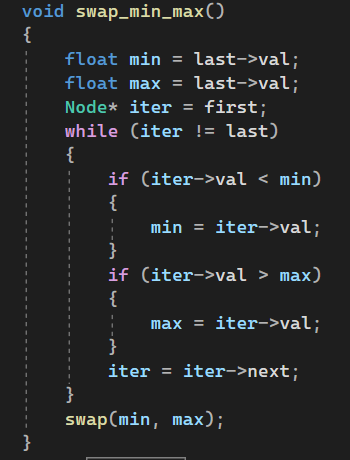


Рисунок 32 – Код метода

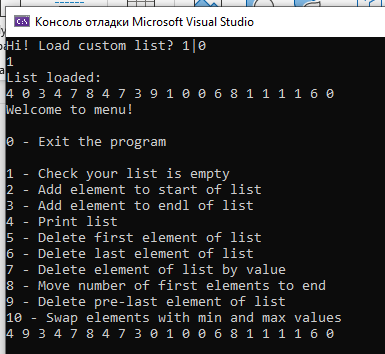


Рисунок 33 – Тест метода

# КОД ВСЕЙ ПРОГРАММЫ

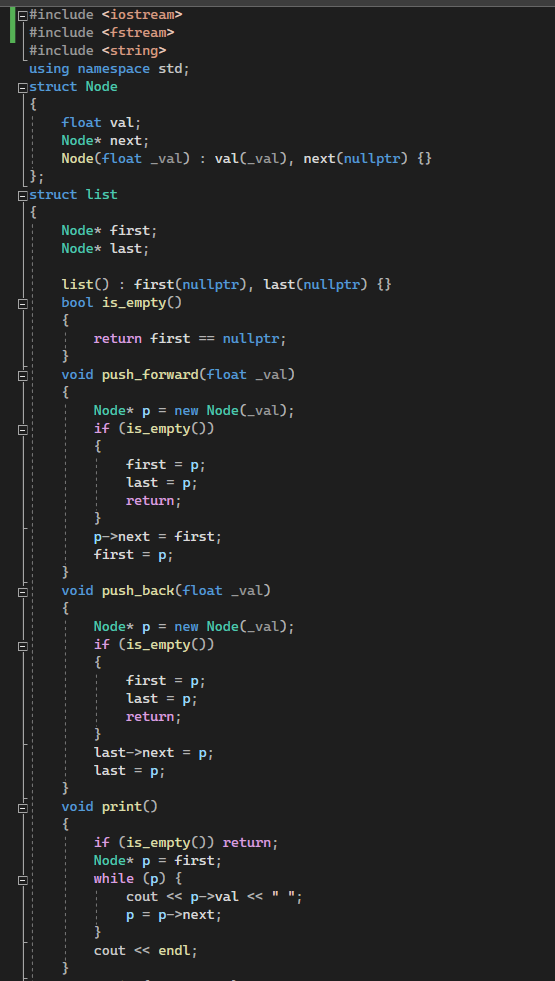


Рис 34 -первая часть кода всей программы

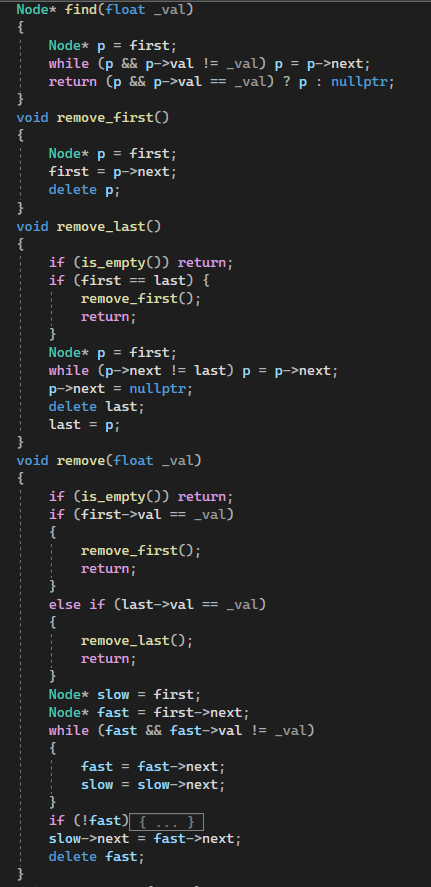


Рис 35 -вторая часть кода всей программы

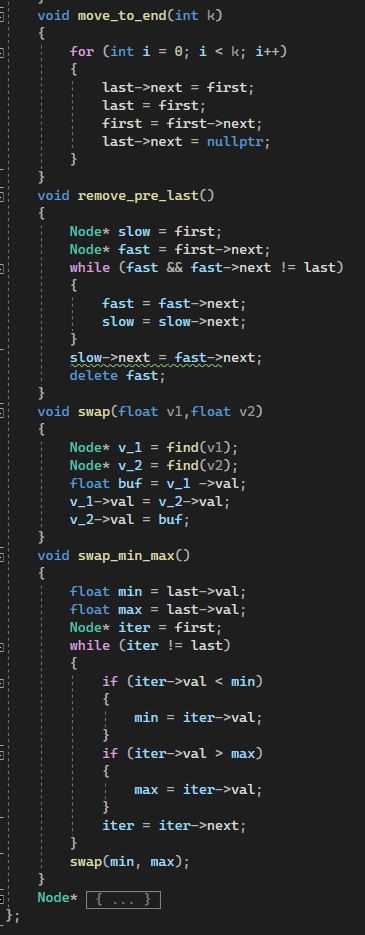


Рис 36 -третья часть кода всей программы

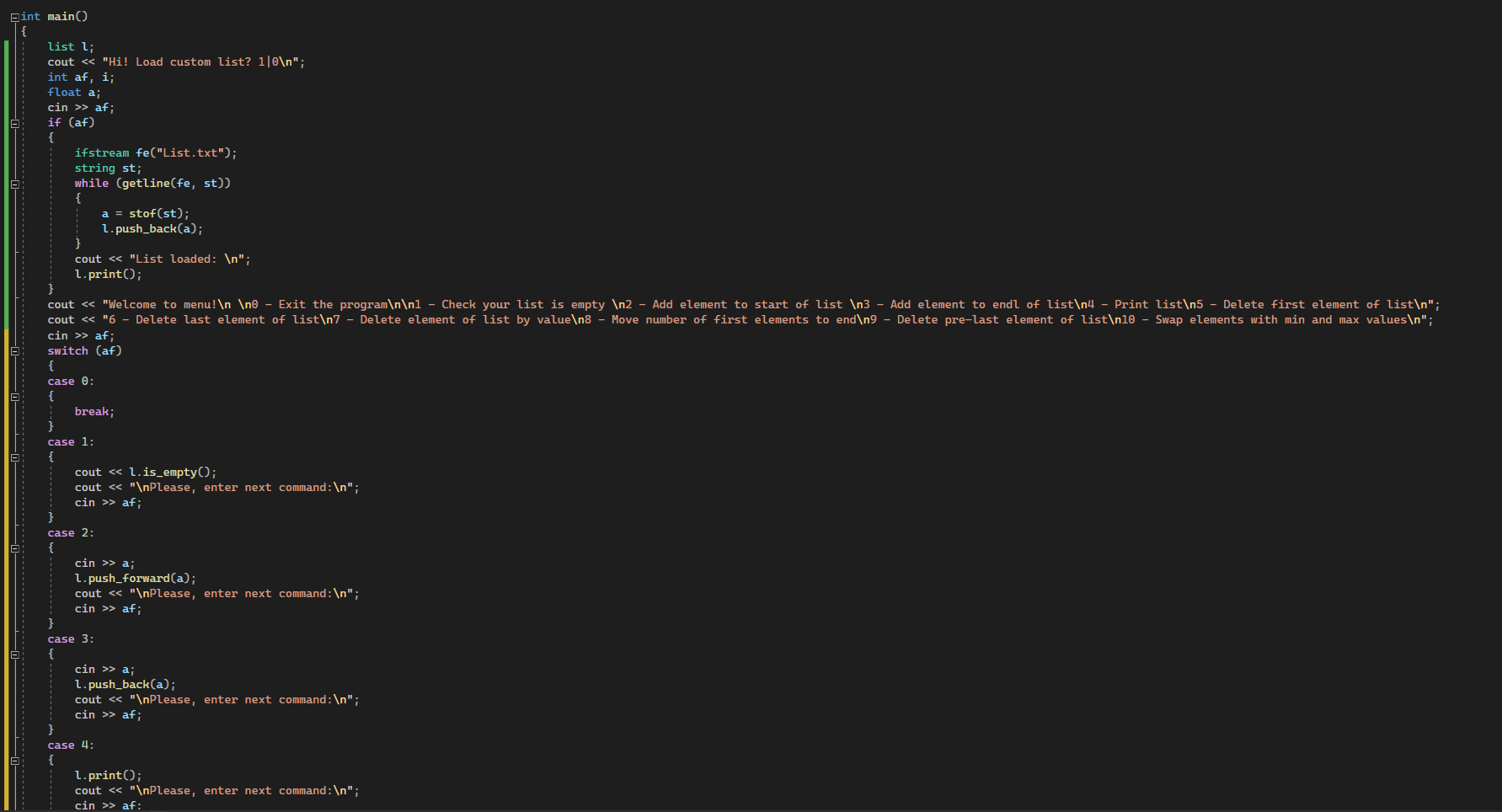


Рис 37 -четвертая часть кода всей программы

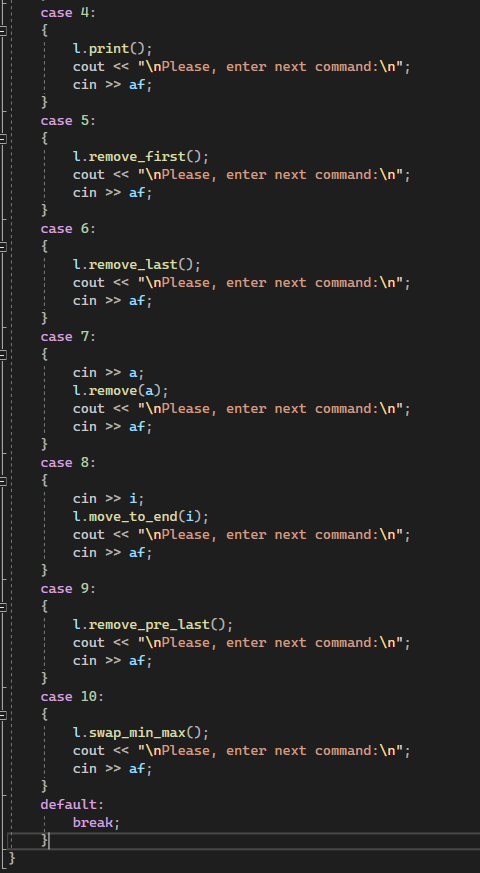


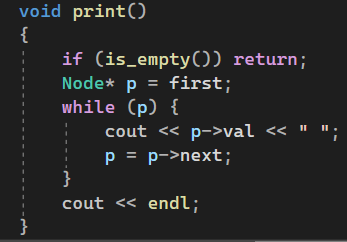
Рис 38 -пятая часть кода всей программы

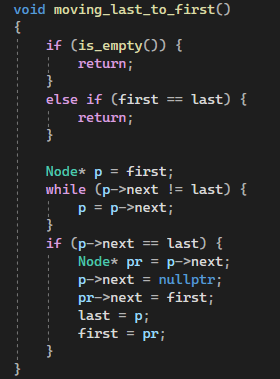
# ВЫВОДЫ

В ходе выполнения практического задания были получены навыки по реализации однонаправленного линейного списка. Выяснилось, как именно работает данная линейная структура хранения многоэлементных последовательностей. Кроме того, были разработаны вспомогательные функции, увеличивающие функционал данной структуры.

# ОТВЕТЫ НА ВОПРОСЫ

Ответим на предложенные вопросы:

1. Существует три уровня представления данных: уровень пользователя  
   (предметная область), логический и физический. Каждый объект предметной области характеризуется своими атрибутами, каждый атрибут имеет имя и значение.
2. Тип данных определяет:  
   - Внутреннее представление данных в памяти компьютера;  
   - Множество значений, которые могут принимать величины этого типа;  
   - Операции и функции, которые можно применять к величинам этого типа.
3. Структура данных — это контейнер, который хранит данные в определенном макете. Этот «макет» позволяет структуре данных быть эффективной в некоторых операциях и неэффективной в других.
4. Структуры хранения данных могут быть статическими или динамическими. Статические структуры создаются при компиляции программы в области данных и их размер изменить нельзя во время работы программы. Они освобождают память, выделенную им, только по завершении программы.  
   Динамические структуры создаются во время работы программы в динамической памяти, их размер может быть изменен во время работы программы. Память этой структуры хранения может и ее отдельных элементов быть освобождена во время работы программы.
5. Линейные [структуры данных представляют список элементов](https://topuch.ru/programma-kompilyator/index.html), упорядоченных по положению. Доступ к элементу прямой по номеру.
6. Структура данных задачи могут представлять линейный список значений – т.е. список значений, следующих друг за другом. Для таких данных можно использовать векторную структуру данных. Вектор представляет линейную структуру, состоящую из нескольких последовательно расположенных ячеек.  
   Ячейка предназначена только для хранения значения и связи со следующей за ней ячейкой она не хранит.
7. Стек — структура данных, представляющая из себя упорядоченный набор элементов, в которой добавление новых элементов и удаление существующих производится с одного конца, называемого вершиной стека.
8. Очередь – это структура данных, представляющая собой последовательность элементов, образованная в порядке их поступления.
9. Список — это линейная цепочка элементов с последовательным доступом и вставкой / удалением элементов в любом месте списка. Стек — это доступ по схеме LIFO: "последним пришёл, первым ушёл", добавление и удаление элементов происходит с одного конца, а не в любом месте. Список - один из способов реализации стека в программе. Но это далеко не единственный и не самый эффективный способ.
10. Для вывода введённой последовательности, наоборот, лучше использовать линейную структуру Дек, так как в ней доступ к элементам осуществляется как с начала, так и с конца..
11. a) Сложность вставки в массив O(n); б) Сложность вставки в структуру O(n).
12. a) Сложность удаления элемента из массива O(n); б) Сложность удаления элемента из структуры O(n).
13. Трюк Вирта заключается в том, чтобы указатель на удаляемый элемент перенаправить на элемент, следующий за удаляемым.
14. Односвязный список состоит из узлов. Узел содержит значение и указатель на следующий узел.
15. Вывод списка на экран  
    
16. Реализация операции перемещения последнего элемента в начало списка



1. Лишним является применение условного оператор if. Для правильности работы задуманной программы следует убрать условный оператор, оставив только фрагмент кода, находящийся в теле оператора else. Тогда созданный узел устанавливается на место между первого и второго узла исходного списка.