МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

**«Нижегородский государственный университет им. Н.И. Лобачевского»**

**Национальный исследовательский университет**

**Институт информационных технологий, математики и механики**

**Кафедра математического обеспечения и суперкомпьютерных технологий**

**Отчет по учебной практике**

**«Структура хранения данных:**

**Односвязный список с использованием массивов»**

**Выполнил:** студент группы 381706-1

Максимова Ирина Игоревна

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

ассистент каф. МОСТ ИИТММ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лебедев И.Г

Нижний Новгород

2018.

Содержание

[1. Введение 3](#_Toc532665011)

[2. Постановка задачи 4](#_Toc532665012)

[3. Руководство пользователя 5](#_Toc532665013)

[4. Руководство программиста 6](#_Toc532665014)

[4.1 Описание структуры программы 6](#_Toc532665015)

[4.2 Описание структур данных 6](#_Toc532665016)

[4.3 Описание алгоритмов 7](#_Toc532665017)

[5. Заключение 8](#_Toc532665018)

[6. Литература 9](#_Toc532665019)

# Введение

Односвязный список на массивах – это структура данных, представляющая собой упорядоченный набор значений, в котором некоторое значение может встречаться более одного раза.

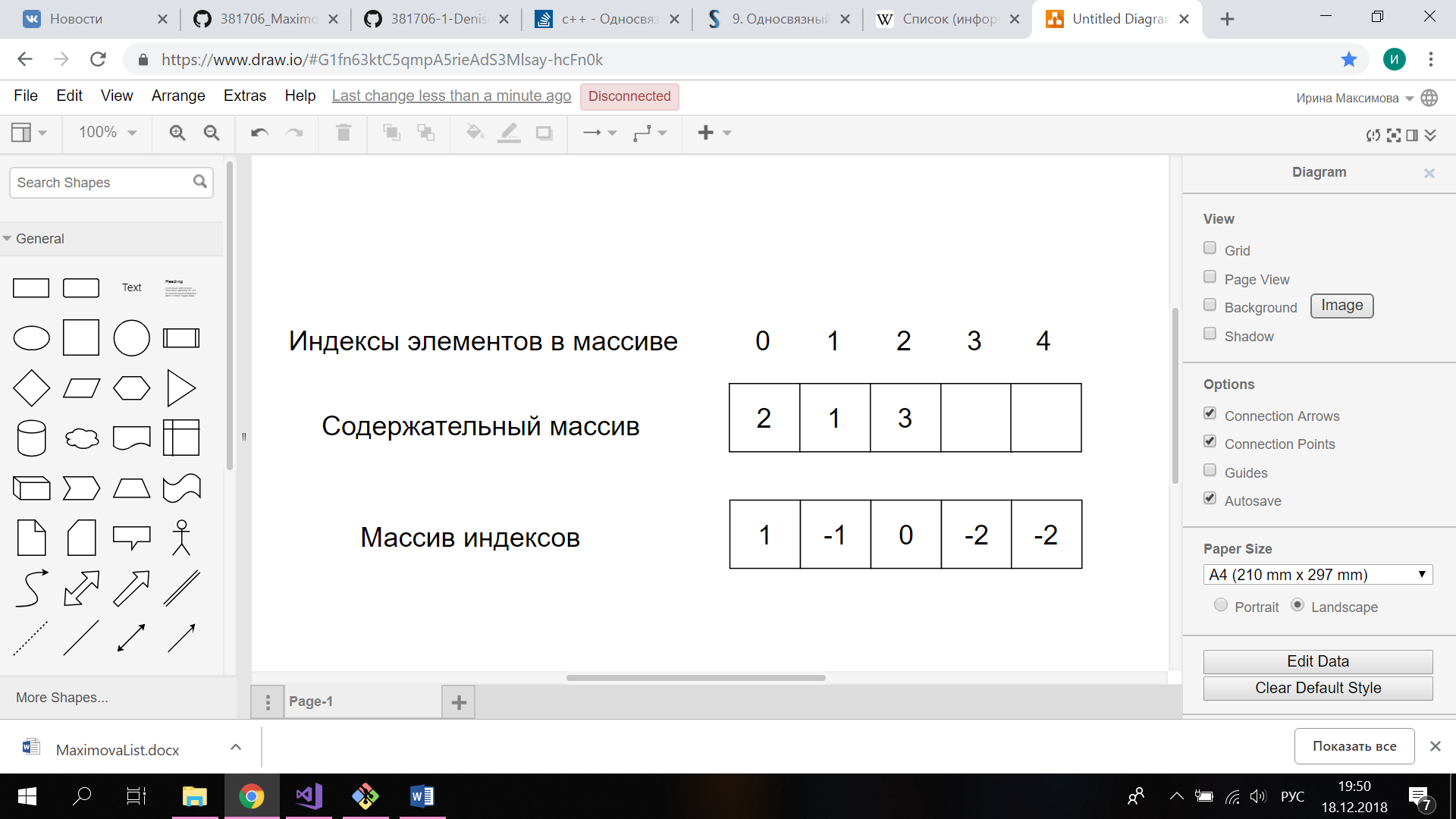
Для создания такого списка используется два массива одинакового размера: содержательный массив и массив индексов. Эти массивы связанны друг с другом по индексу. По фиксированному *i*, содержательный массив, в *i*-й ячейке, хранит значение элемента списка, а массив индексов, в *i*-й ячейке, содержит индекс следующего элемента списка в содержательном массиве.

Если список заполнен не полностью, то для пустых ячеек в содержательном массиве, в соответствующих ячейках в массиве индексов лежит значение «-2». Для последнего элемента списка, в соответствующей ячейке массива индексов лежит значение -1.

Как правило, стоит дополнительно хранить индекс первого элемента списка в содержательном массиве.

Рассмотрим пример.

Мы имеем список целых чисел из 3-х элементов. Всего список может содержать 5 элементов. Мы знаем, что первый элемент в списке - «3». В памяти компьютера список выглядит как на рис 1. Следовательно можем сделать вывод что искомый список - «3, 2, 1».



Лабораторная работа направлена на практическое освоение структуры данных – линейного односвязного списка на массивах.

# Постановка задачи

В рамках лабораторной работы ставится задача эффективной реализации структуры данных – линейного односвязного списка на массивах.

Для работы со списком необходимо реализовать операции:

* Добавления элемента в начало списка,
* добавления элемента в конец списка,
* извлечения с удалением элемента из начала списка,
* извлечения с удалением элемента из конца списка,
* проверка списка на полноту/пустоту.

Программное решение будет выглядеть следующим образом:

1. Класс списка – TArrList.
2. Класс для обработки исключений – MyException, которые могут возникнуть при выполнении различных операций.
3. Программа, демонстрирующая работу класса TArrList.
4. Набор автоматических тестов с использованием Google C++ Testing Framework.

# Руководство пользователя

Рассмотрим пример использования класса TArrList.

При запуске программы с пользователя спрашивается размер списка . Затем в начало списка кладется Затем спрашивается количество элементов , которое необходимо положить в конец списка. Полученный список выводится на экран. Он будет выглядеть следующим образом: .

Затем выполняется изъятие элементов из конца и начала списка. Изъятые элементы выводятся на экран. В конце на консоль выводится список, полученный после изъятия из него элементов. На этом работа программы прекращается.

# Руководство программиста

## 4.1 Описание структуры программы

Программа состоит из следующих модулей:

* Модуль ArrList. Содержит пример использования списка на массивах. Реализация в файле *main\_ArrList.cpp.*
* Модуль ArrListLib – статическая библиотека. Содержит файл ArrList.h, в котором описан интерфейс и реализация шаблонного класса *TArrList*.
* Модуль ArrListTest. Содержит 17 тестов, описанных в файле *ArrListTest.cpp* и разработанных с помощью использования Google C++ Testing Framework.
* Модуль ExceptionLib – библиотека, позволяющая создавать собственные исключения.

## Описание структур данных

#### Класс TArrList – класс списка на массивах.

Класс TArrList является шаблонным. В классе всего 8 полей, объявленных со спецификатором доступа private:

* T\* mas – указатель на область памяти для хранения элементов списка.
* int \*nextInd - указатель на область памяти для хранения индексов, указывающих на следующий элемент списка.
* int \*predInd - указатель на область памяти для хранения индексов, указывающих на предыдущий элемент списка.
* int size – максимальный размер списка.
* int count – текущее количество элементов в списке.
* int start - индекс первого элемента списка.
* int end - индекс последнего элемента списка.
* TQueue <int> freeElem - Очередь свободный ячеек в массиве mas.

**Конструкторы и методы класса, объявленные со спецификатором public:**

* TArrList(int \_size = 10) - конструктор по умолчанию.
* TArrList(TArrList<T> &A) - конструктор копирования.
* ~ TArrList() – деструктор.
* void PutStart(T elem) – метод, позволяющий добавить элемент в начало списка.
* void PutEnd(T elem) – метод, позволяющий добавить элемент в конец списка.
* T GetStart() – метод, позволяющий получить с удалением элемент из начало списка.
* T GetEnd() – метод, позволяющий получить с удалением элемент из конца списка.
* bool IsFull() - проверка списка на полноту.
* bool IsEmpty() – проверка списка на пустоту.
* void Print() – вывод элементов списка на консоль.

## Описание алгоритмов

**Добавление звена списка в начало и в конец.**

Сначала проверяем не заполнен ли список. Если он заполнен, то бросаем исключение. Если нет, то в очереди свободных позиций *freeElem*, берем первую свободную ячейку *i*. По полученному индексу в массив *mas* записываем значение, которое хотим положить в список. Определяем, что следующим для этого элемента, будет элемент с текущим индексом start, то есть *nextInd[i] = start*. Если, перед добавлением, список не был пуст, то предыдущим для первого элемента списка делаем только что добавленный элемент, то есть: *predInd[start] = i.* Если же список был пуст, то определяем, что добавленный элемент является и последним элементом в списке. Затем индекс *start* переопределяем на только что добавленный элемент: *start = i*. Увеличиваем количество элементов в списке *count++*.

Для добавления элемента в конец списка, рассуждения аналогичны, с поправкой на то, что добавляем в конец списка.

**Удаление звена списка из начала и из конца.**

Сначала проверяем не пуст ли список. Если он пуст, то бросаем исключение. Если нет, то в переменную *elem* записываем элемент в начале списка *mas[start].* Затем переопределяем индекс первого элемента списка, он будет равен индексу элемента следующим за первым: *newstart = nextInd[start].* Затем в очередь свободных позиций *freeElem*, добавляем освободившуюся после изъятия первого элемента свободную ячейку *start*. Идентифицируем пустыми соответствующие ячейки массивов *nextInd* и *predInd*: *nextInd[start] = predInd[start]* = -2. Если после извлечения первого элемента, список не пуст, то предыдущего элемента для нового первого элемента не существует: *predInd[newstart] = -1*.Затем индекс *start* переопределяем на *newstart*: *start = newstart*. Уменьшаем количество элементов в списке *count--*.

Для извлечения элемента из конца списка, рассуждения аналогичны.

# Заключение

В ходе выполнения лабораторной был произведен анализ задачи - установлено понятие списка на массивах. Была разработана библиотека, реализующая шаблонный класс списка. Она позволяет при работе со списком выполнять базовые операции извлечения/добавление элементов списка.

Предоставлено описание примера работы со списком в разделе «Руководство пользователя».

Также разработаны и доведены до успешного выполнения тесты, проверяющие корректность методов классов TArrList.

# Литература

* Книги

1. A.O. Грудзинский. Методы программирования, Издательство Нижегородского госуниверситета, 2006.
2. Васильев А.Н. Самоучитель С++ с примерами и задачами. -СПб.: Наука и Техника, 2016. -480с.

* Ссылки в Internet

1. Википедия: свободная электронная энциклопедия: на русском языке: [https://ru.wikipedia.org/wiki/Список\_(информатика)]
2. Гергель В.П. Методические материалы по курсу «Методы программирования 2»: [ http://www.itmm.unn.ru/files/2018/11/Primer-1.7.-Razrabotka-obshhego-predstavleniya-linejnogo-spiska.pdf], 2015.