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 очереди»**

**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](#_Toc533083407)

[2. Постановка задачи 4](#_Toc533083408)

[3. Руководство пользователя 5](#_Toc533083409)

[4. Руководство программиста 6](#_Toc533083410)

[4.1. Описание структуры программы 6](#_Toc533083411)

[4.2. Описание структур данных 6](#_Toc533083412)

[4.3. Описание алгоритмов 6](#_Toc533083413)

[5. Заключение 8](#_Toc533083414)

[6. Литература 9](#_Toc533083415)

# Введение

**Очередь** - структура данных, представляющая собой упорядоченный набор элементов, с дисциплиной доступа к элементам «первый пришёл — первый вышел» (англ. *first-in, first-out — FIFO*). У очереди имеется голова и хвост. Когда элемент ставится в очередь, он занимает место в её хвосте. Из очереди всегда выводится элемент, который находится в ее голове.

Очередь в программировании используется, как и в реальной жизни, когда нужно совершить какие-то действия в порядке их поступления, выполнив их последовательно.

**Цель данной лабораторной работы** – разработка структуры хранения очереди.

# Постановка задачи

Выполнение работы предполагает решение следующих задач:

1. Разработка и реализация класса очередь – TQueue.
2. Создание класса для обработки исключений – MyException, которые могут возникнуть при выполнении различных операций.
3. Реализация примера использования программного решения, демонстрирующее работу класса TQueue.
4. Создание набора автоматических тестов с использованием Google C++ Testing Framework.

# Руководство пользователя

Рассмотрим пример использования класса TQueue.

При запуске программы c пользователя спрашивается размер очереди . Затем создается очередь и полностью заполняется натуральными числами от 0 до . Каждое добавленное число в очередь выводится на консоль.

Далее, пока очередь не пуста, из нее извлекаются все элементы. И в порядке извлечения выводятся на экран. На этом работа программы прекращается.

# Руководство программиста

## Описание структуры программы

Программа состоит из следующих модулей:

* Модуль Queue. Содержит пример использования очереди. Реализация в файле *main\_Queue.cpp.*
* Модуль QueueLib – статическая библиотека. Содержит файл Queue.h, в котором описан интерфейс и реализация шаблонного класса *TQueue* (2 конструктора, деструктор и 4 метода).
* Модуль StackTest. Содержит 10 тестов, описанных в файле *StackTest.cpp* и разработанных с помощью использования Google C++ Testing Framework.
* Модуль ExceptionLib – библиотека, содержащая класс исключений.

## Описание структур данных

#### 4.2.1 Класс TQueue

Класс *TQueue* является шаблонным классом, причем он наследуется от класса TStack. В нем определены два поля со спецификатором доступа protected:

* *int start* – позиция вершины стека.
* *Int count* – количество, заполненных элементов очереди.

Далее в публичной зоне (public):

1. Прописаны 2 конструктора и деструктор:

* *TQueue(int n = 0)* – конструктор по умолчанию.
* *TQueue(TQueue<T> &q)* – конструктор копирования.
* *virtual ~TQueue()* – деструктор.

1. Перегружены методы для работы с очередью:

* *void Put(T A)* – добавить новый элемент A в конец очереди.
* *T Get()* – изъять с удалением элемент находящийся вначале очереди.
* *bool IsFull()* – проверка очереди на полноту.
* *bool IsEmpty()* – проверка очереди на пустоту.

## Описание алгоритмов

**Добавление элемента в очередь.**

При добавлении элемента в очередь, размещаем его в свободную ячейку массива на которую указывает *top* – вершина очереди (если она имеется). Поскольку после выхода из очереди хотя бы одного элемента, массив, в котором она хранится будет выглядеть как на рисунке 1, значение *top* рассчитывается по следующей формуле:

|  |  |
| --- | --- |
|  |  |

Такая реализация поиска последнего элемента в очереди необходима, чтобы в случае, изображенном на рисунке 1, можно было заполнить все свободные ячейки в массиве, в котором хранится очередь.

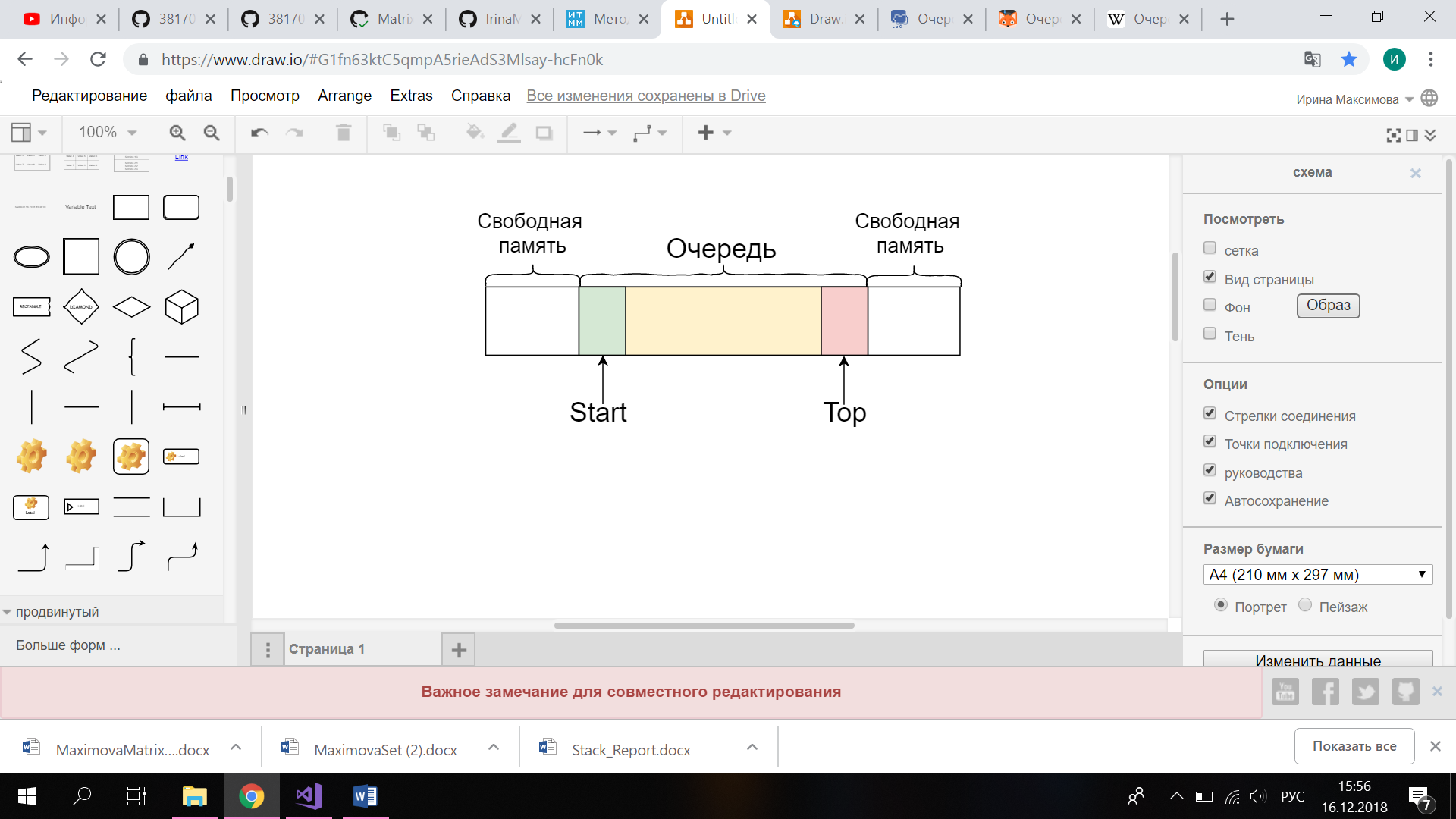


Рисунок 1. Представление очереди в памяти компьютера на n-й операции добавления и изъятия элементов.

**Удаление элемента из очереди.**

При изъятии элемента из очереди, возвращается элемент с индексом начала очереди *start*, а сам индекс *start* рассчитывается по формуле:

|  |  |
| --- | --- |
|  |  |

Такая реализация поиска следующего начального элемента очереди необходима, чтобы в случае, изображенном на рисунке 2, можно было забрать все элементы очереди.

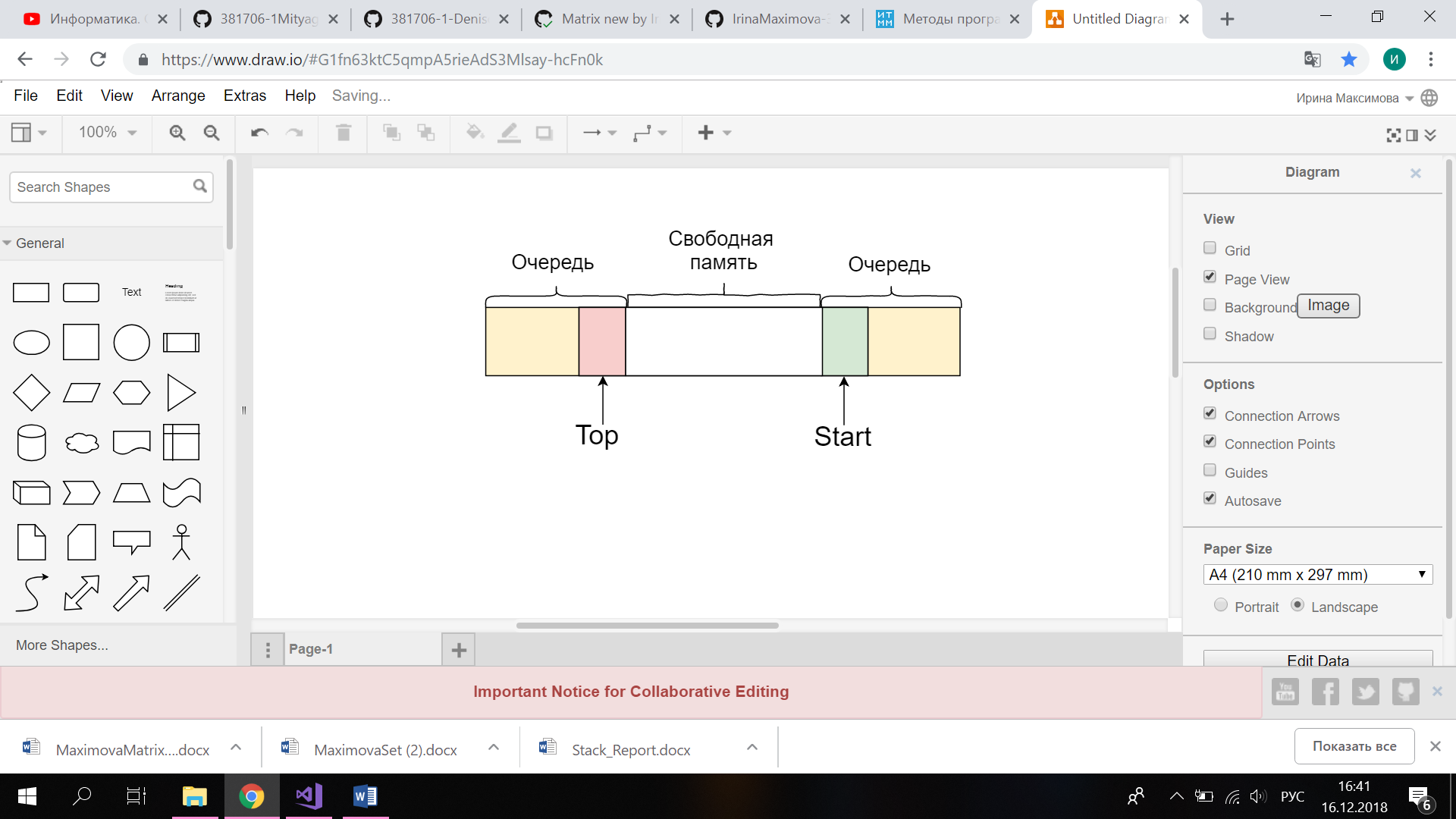


Рисунок 2. Представление очереди в памяти компьютера на n-й операции добавления и изъятия элементов.

Само значения изъятого элемента не затирается, т.к. оно будет перезаписано при следующем добавлении другого элемента.

# Заключение

В ходе выполнения лабораторной был произведен анализ задачи - установлено понятие очереди. Была разработана библиотека QueueLib, реализующая шаблонный класс стека TQueue. В ней реализованы методы работы с очередью описанные в разделе «Структуры данных».

Программная реализация очереди была продемонстрирована на примере, описывающем основные методы класса TQueue.

Разработаны и доведены до успешного выполнения тесты, разработанные для данного программного проекта с использованием Google C++ Testing Framework.

# 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. Гергель В.П. Методические материалы по курсу «Методы программирования 2»: [http://www.itmm.unn.ru/files/2018/10/Primer-1.4.-Struktury-hraneniya-ocheredi.pdf], 2015.
2. Википедия: свободная электронная энциклопедия: на русском языке:

[https://ru.wikipedia.org/wiki/Очередь\_(программирование)]