МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Национальный исследовательский**

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

**(ННГУ)**

**Институт информационных технологий, математики и механики**

**Кафедра математического обеспечения и суперкомпьютерных технологий**

**Отчет по учебной практике**

**«Структура хранения данных: Очередь»**

**Выполнил:** студент группы 381706-1

Колесова Кристина Юрьевна

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

ассистент каф. МОСТ ИИТММ

Лебедев Илья Геннадьевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись

Нижний Новгород

2018.

Оглавление

[1. Введение 3](#_Toc534392569)

[2. Постановка задачи 4](#_Toc534392570)

[3. Руководство пользователя 5](#_Toc534392571)

[4. Руководство программиста 6](#_Toc534392572)

[4.1. Описание структуры программы 6](#_Toc534392573)

[4.2. Описание структур данных 6](#_Toc534392574)

[4.3. Описание алгоритмов 7](#_Toc534392575)

[5. Заключение 9](#_Toc534392576)

[6. Список литературы 10](#_Toc534392577)

# Введение

Очередь – абстрактный тип данных с дисциплиной доступа к элементам «первый пришёл — первый вышел» (FIFO, англ.first in, first out). Добавление элемента возможно лишь в конец очереди, извлечение – только из начала очереди, при этом выбранный элемент из очереди удаляется.

Цель данной лабораторной работы – реализация структуры данных очередь.

# Постановка задачи

Выполнение работы предполагает решение следующих задач:

1. Реализация класса TQueue, на основе класса TStack;
2. Реализация класса для обработки исключений – Texception;
3. Реализация тестов на базе Google Test;
4. Пример использования класса TQueue.

В классе TQueue предполагается реализовать следующие методы:

* + 1. Put – добавление элемента;
    2. Get – удаление элемента;
    3. IsEmpty – проверка очереди на пустоту;
    4. IsFull – проверка очереди на полноту.

# Руководство пользователя

В качестве примера использования очереди предлагается следующая тестовая программа.

1. Пользователю предлагают ввести количество элементов в очереди, после чего создается стек заданного размера для хранения целочисленных значений;
2. Пользователю предлагают заполнить ячейки очереди целыми значениями;
3. Заданная очередь выводится на консоль;
4. Удаляется один элемент из очереди;
5. Производится вывод очереди на консоль;
6. С помощью конструктора копирования создается копия очереди;
7. Пользователю предлагается ввести новый элемент исходной очереди;
8. Дополненная очередь выводится на консоль;
9. На консоль выводится копия очереди до добавления последнего элемента.

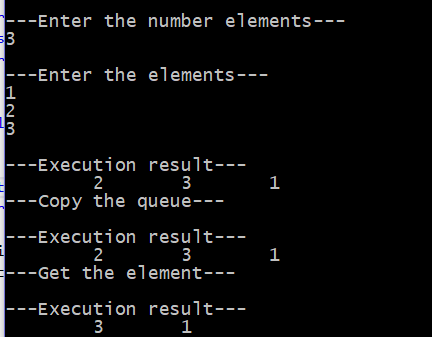
На этом работа программы заканчивается.

Рисунок . Пример работы программы

# Руководство программиста

## Описание структуры программы

В программе содержатся следующие модули:

* Модуль Queue - содержит файл main.cpp, в котором реализован пример использования очереди.
* Модуль QueueLib –содержит файл Queue.h, в котором описан интерфейс и реализация шаблонного класса *TQueue*.
* Модуль QueueTest – содержит тесты для класса TQueue, разработанные с помощью использования фреймворка Google Test.
* Модуль Exception – содержит файл Exception.h, в котором реализован класс исключений TException.

## Описание структур данных

* Класс TQueue

Класс TQueue является шаблонным. Также класс TQueue наследуется от ранее разработанного класса TStack..

1. Наследуются из класса TStack:

int size – размер очереди.

int top – индекс ячейки в массиве, в которой расположен элемент на выход.

T\* mas – указатель на область памяти, выделенную под хранение элементов очереди

1. Элементы класса, объявленные со спецификатором protected:

int start – индекс ячейки в массиве, в которую можно добавить очередной элемент.

int count – количество элементов в очереди.

1. Элементы класса, объявленные со спецификатором public:

TQueue(int n = 0) – конструктор с одним параметром.

TQueue(TQueue <T> &q) – конструктор копирования.

virtual ~TQueue() – деструктор.

void Put(T a) – метод, позволяющий добавить новый элемент в очередь.

T Get() – метод изъятия элемента из очереди с удалением.

void Print() – метод отображения текущих элементов очереди.

bool IsFull() – метод проверки очереди на полноту.

bool IsEmpty() – метод проверки очереди на пустоту

* Класс TException

Класс содержит 1 private поле:

string msg – переменная, хранящая сообщение об ошибке в виде строки.

Содержит 2 public элемента:

msg(\_msg) – конструктор.

void Print() – метод отображения ошибки на консоль.

## Описание алгоритмов

* Добавление элемента в очередь

При добавлении элемента в очередь обращаемся к элементу массива с индексом start, записываем туда полученное значение элемента a. Значение start увеличиваем согласно следующей формуле . Такое вычисление индекса необходимо из-за того, что для хранения очереди мы используем не просто массив, а кольцевой буфер. Увеличиваем на 1 число элементов в очереди count.

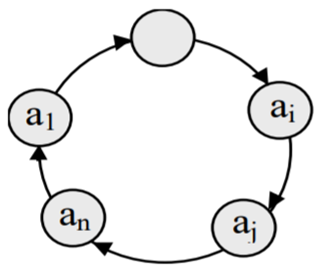


Рисунок 2. Кольцевой буфер

* Удаление элемента из очереди

При удалении элемента из очереди забираем значение элемента массива с индексом *top*. Значение *top* увеличивается согласно следующей формуле: . Уменьшаем на 1 число элементов в очереди *count.*

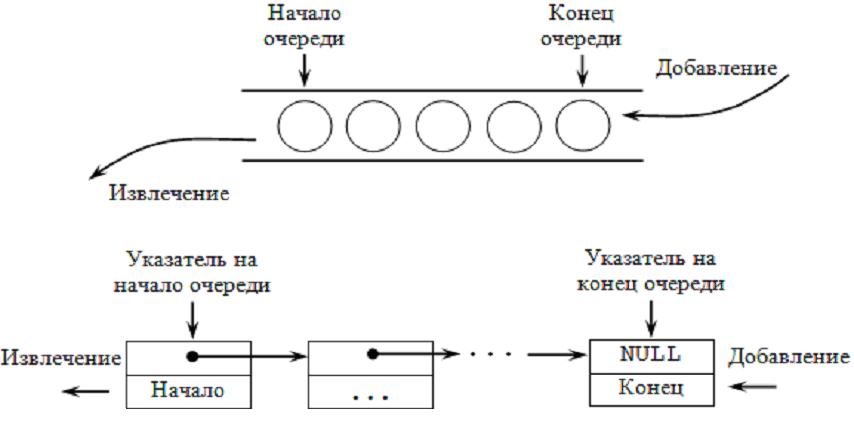


Рисунок 3. Добавление и удаление элемента очереди

# Заключение

В результате выполнения лабораторной работы была разработана библиотека, позволяющая работать с шаблонным классом очереди. С помощью этой библиотеки можно создать объект класса очереди и выполнить с ним операции добавления, извлечения, проверки на пустоту и полноту.

Были реализованы тесты для проверки работоспособности класса стека на базе GoogleTest. А также приведен пример, демонстрирующий работу с очередью.

# Список литературы

1. Википедия. Статья «Очередь (программирование)»: [https://ru.wikipedia.org/wiki/Очередь\_(программирование)].
2. Васильев А.Н. Самоучитель С++ с примерами и задачами. -СПб.: Наука и Техника, 2016. -480с.
3. Гергель В.П. Методические материалы по курсу «Методы программирования 2», Нижний Новгород, 2015.