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 данных: Очередь»**

**Выполнил:** студент группы 381706-1

Власов Андрей Сергеевич

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

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

Нижний Новгород

2018.

Содержание

[1. Введение 3](#_Toc532752145)

[2. Постановка задачи 4](#_Toc532752146)

[3. Руководство пользователя 5](#_Toc532752147)

[4. Руководство программиста 6](#_Toc532752148)

[4.1 Описание структуры программы 6](#_Toc532752149)

[4.2 Описание структур данных 6](#_Toc532752150)

[4.3 Описание алгоритмов 7](#_Toc532752151)

[5. Заключение 8](#_Toc532752152)

[6. Литература 9](#_Toc532752153)

# Введение

Очередь — структура данных, представляющая из себя упорядоченный набор элементов, в которой доступ к элементам производится по принципу «первым вошел — первым вышел» (англ. *first-in, first-out — FIFO*). У очереди имеется голова и хвост. Когда элемент ставится в очередь, он занимает место в её хвосте. А выводится элемент, который находится в её голове.

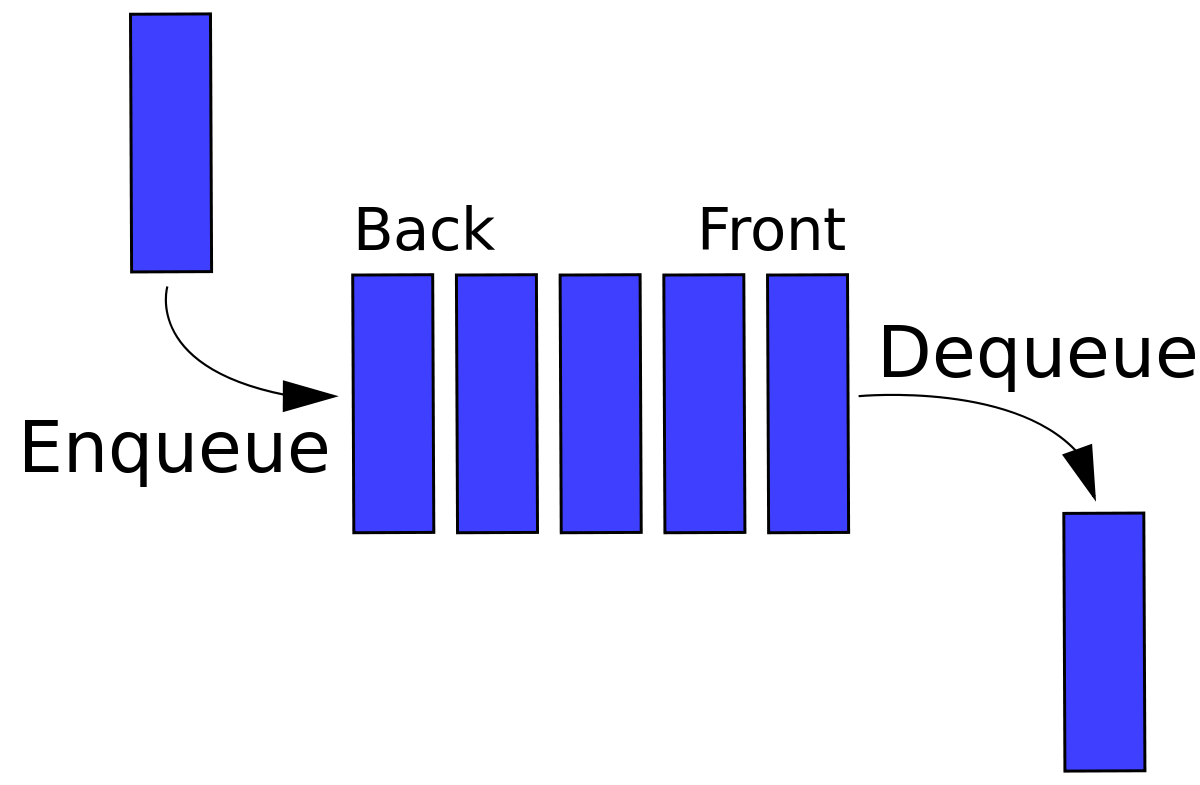


Рисунок 1 Схема очереди.

Цель данной лабораторной работы: реализация очереди.

# Постановка задачи

В данной лабораторной работе нам необходимо осуществить:

1. Разработку и реализацию класса очереди.
2. Создание класса для обработки исключений.
3. Пример использования класса очереди.
4. Набор автоматических тестов.

# Руководство пользователя

Рассмотрим пример написания глобальной функции main:

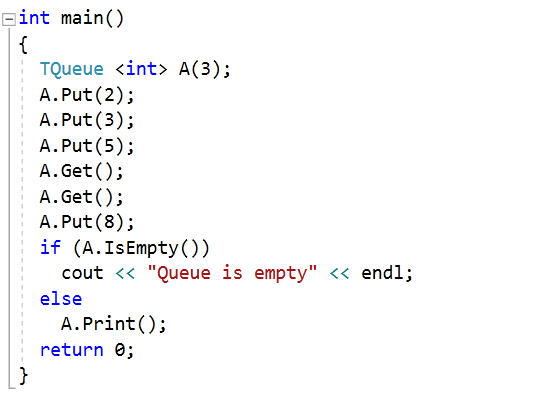


Рисунок 2 Пример использования класса TQueue

В данном случае мы создаем очередь с максимальным размером 3, добавляем и удаляем элементы, производим проверку на пустоту и выводим результат.

# Руководство программиста

## 4.1 Описание структуры программы

Программа состоит из следующих модулей:

* Модуль Queue. Содержит пример использования стека.
* Модуль StackLib – статическая библиотека. Содержит файл Stack.h, в котором описан интерфейс и реализация шаблонного базового класса *TStack*.
* Модуль Queue Lib – статическая библиотека. Содержит файл Queue.h, в котором описан интерфейс и реализация шаблонного класса TQueue, наследуемого от *TStack.*
* Модуль Queue Test. Содержит набор тестов, разработанных с помощью использования Google C++ Testing Framework.
* Модуль MyExceptionsLib – библиотека исключений.

## Описание структур данных

#### Класс TQueue – очередь.

template <class T> class TQueue : protected TStack<T> – класс очереди – наследник класса стека.

Он содержит 2 новых поля со спецификатором доступа protected:

1. int start – позиция головы очереди.
2. int count – количество элементов в очереди.

А также 2 конструктора, 1 деструктор и 5 методов со спецификатором доступа public:

* TQueue(int \_size) – конструктор инициализации.
* TQueue(TQueue <T> &A) – конструктор копирования.
* ~TQueue() – деструктор.
* void Put(T e) – метод добавления нового элемента в очередь.
* T Get() – метод изъятия элемента из очереди с удалением.
* void Print() – метод вывода элементов очереди.
* bool IsFull() – метод проверки очереди на полноту.
* bool IsEmpty() – метод проверки очереди на пустоту.

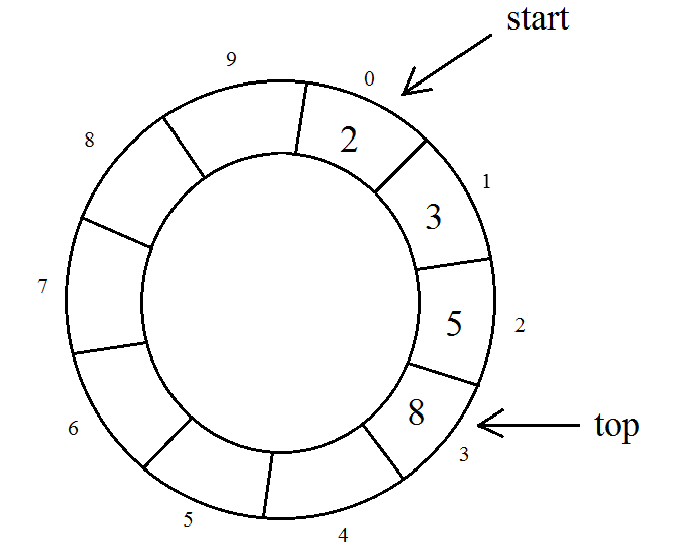
## Описание алгоритмов

**Добавление элемента в очередь.**

Сначала проверяем очередь на полноту, если она неполная, то размещаем элемент в ячейку массива с индексом top. Значение top изменяем по формуле top = (top + 1) % size. Обусловлено это тем, что наша очередь - это кольцевой буфер. Увеличиваем число элементов count на один.

**Удаление элемента из очереди.**

Сначала проверяем очередь на пустоту, если она непустая, то возвращаем элемент с индексом начала очереди start. Значение start изменяем по формуле start = (start + 1) % size. Уменьшаем число элементов count на один.



*Рисунок 3 Представление очереди.*

# Заключение

В данной лабораторной работе была выполнена реализация такой структуры данных как очередь. А именно была разработана библиотека, реализующая шаблонный класс очереди, который наследуется от базового класса стека, был создан класс для обработки исключений, показан пример использования, а также написан набор тестов с использованием Google C++ Testing Framework.

# Литература

Internet – ресурсы:

1. <https://ru.wikipedia.org/wiki/Очередь_(программирование)>
2. <https://neerc.ifmo.ru/wiki/index.php?title=Очередь>

Книги:

1. Гергель В.П. Методические материалы по курсу «Методы программирования 2», Нижний Новгород, 2015.
2. Ахо А., Хопкрофт Д., Ульман Д. Структуры данных и алгоритмы.: Пер. с англ. М.: Издат. дом «Вильямс», 2000. С. 58–76