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**

**«Очередь»**

**Выполнил:** студент группы 381706-1

Суслов Егор Игоревич

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

ассистент каф. МОСТ ИИТММ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лебедев И.Г

Нижний Новгород

2018.

**Содержание**

[**1.** **Введение** 3](#_Toc510437279)

[**2.** **Постановка задачи** 4](#_Toc510437280)

[**3.** **Руководство пользователя** 5](#_Toc510437281)

[**4.** **Руководство программиста** 6](#_Toc510437282)

[**4.1.** **Описание структуры программы** 6](#_Toc510437283)

[**4.2.** **Описание структур данных** 6](#_Toc510437284)

[**5.** **Заключение** 8](#_Toc510437285)

[**6.** **Литература** 9](#_Toc510437286)

# **Введение**

Очередь (англ. queue)  — это структура данных, добавление элемента в которой (принято обозначать словом enqueue — поставить в очередь) возможно лишь в конец очереди, выборка — только из начала очереди (что принято называть словом dequeue — убрать из очереди), при этом выбранный элемент из очереди удаляется.. Притом первым из очереди удаляется элемент, который был помещен туда первым, то есть в очереди реализуется принцип «первым вошел — первым вышел» (англ. first-in, first-out — FIFO). У очереди имеется голова (англ. head) и хвост (англ. tail). Когда элемент ставится в очередь, он занимает место в её хвосте.

Очередь характеризуется таким порядком обработки значений, при котором вставка новых элементов производится в конец очереди, а извлечение – из начала. Подобная организация данных широко встречается в различных приложениях. В качестве примера использования очереди предлагается задача разработки системы имитации однопроцессорной ЭВМ. Рассматриваемая в рамках лабораторной работы схема имитации является одной из наиболее простых моделей обслуживания заданий в вычислительной системе и обеспечивает тем самым лишь начальное ознакомление с проблемами моделирования и анализа эффективности функционирования реальных вычислительных систем.

Таким образом, очередь есть динамическая структура, операции вставки и удаления переводят очередь из одного состояния в другое, при этом добавление новых элементов осуществляется в конец очереди, а извлечение – из начала очереди (дисциплина обслуживания «первым пришел – первым обслужен.

Важной задачей при реализации системы обслуживания очереди является выбор структуры хранения, обеспечивающей решение проблемы эффективного использования памяти без перепаковок и без использования связных списков (требующих дополнительных затрат памяти на указатели).

Как и в случае со стеком, в качестве структуры хранения очереди предлагается использовать одномерный (одноиндексный) массив, размещаемый в динамической области памяти. В связи с характером обработки значений, располагаемых в очереди, для указания хранимых в очереди данных необходимо иметь два указателя – на начало и конец очереди. Эти указатели увеличивают свое значение: один при вставке, другой при извлечении элемента.

Таким образом, в ходе функционирования очереди может возникнуть ситуация, когда оба указателя достигнут своего наибольшего значения и дальнейшее пополнение очереди станет невозможным, несмотря на наличие свободного пространства в очереди. Одним из решений проблемы «движения» очереди является организация на одномерном массиве кольцевого буфера. Кольцевым буфером называется структура хранения, получаемая из вектора расширением отношения следования парой p(an,a1).

# **Постановка задачи**

Реализовать класс TQueue для работы с данными различных типов используя класс стек TStack . Для реализации использовать шаблоны.

Тип полей:

int .

Обязательно должны присутствовать методы:

* Добавить элемент в очередь
* Удалить элемент из очереди

Должны быть реализованы конструкторы: по умолчанию, копирования и инициализатор.

# **Руководство пользователя**

1. После запуска программу пользователю предлагается ввести 5 элементов очереди
2. Затем введенные числа выводятся на экран
3. Выход

# **Руководство программиста**

## **Описание структуры программы**

Данная программа содержит модули:

* Queue содержит пример использования очереди.

Реализация в файле *«*Queue\_main.cpp».

* QueueLib – статическая библиотека. Содержит файл «Queue.h», в котором описана реализация шаблонного класса TQueue.
* StackLib - библиотека StackLib содержит модуль «Stack.h» описывающий класс шаблонный класс TStack.
* MyExceptionLib – библиотека c классом исключений
* QueueTest c Google Tests в файле «QueueTest.h»
  1. **Описание структур данных**

Описывающий шаблонный класс наследуемый от стека:

Класс TQueue : public TStack <T>.

Содержит поля с идентификатором protected :

* int Start; - начало очереди
* int Count;  - конец очереди

Содержит методы:

1. TQueue(int n = 0); - по умолчанию размер очереди равен 0
2. TQueue(TQueue <T> &Q); - конструктор копирования
3. virtual ~TQueue(); - деструктор
4. void Put(T A); - добавить элемент в очередь
5. T Get(); - убрать элемент из очереди
6. bool IsFull(); - проверка на полноту
7. bool IsEmpty() – проверка на пустоту
8. void Print(); - вывод очереди
9. T Top(); - вершина очереди

# 

# **Заключение**

В ходе работы мне удалось изучить:

1. Специфику типа хранения данных очередь
2. Методы очереди
3. Применение очереди
4. Использование методов стека в очереди

Написанная мною библиотека удачно выполняет следующие действия с данными любого типа:

1. Поместить элемент в очередь;
2. Удалить элемент из очереди;

# 

# **Литература**

* Книги:

1. A.O. Грудзинский. Методы программирования, Издательство Нижегородского госуниверситета, 2006.
2. Топп У., Форд У. Структуры данных в С++. - М. Бином, 1999.
3. Мейн М., Савитч У. Структуры данных и другие объекты в С++. - М.: Издательский дом “Вильямс”, 2003.
4. Гергель В.П. Методические материалы по курсу «Методы программирования 2», Нижний Новгород, 2015.

* Ссылки в Internet:

1. Учебно-методическое пособие из электронной библиотеки ННГУ: «ЛАБОРАТОРНЫЙ ПРАКТИКУМ» : [http://www.unn.ru/books/met\_files/Pract\_ADS.pdf].