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: Стек на списках»**

**Выполнил:** студент группы 381706-1

Шашкин Евгений Вадимович

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

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

Нижний Новгород

2018.

Оглавление

[1. Введение. 2](#_Toc1249355)

[2. Постановка задачи. 3](#_Toc1249356)

[3. Руководство пользователя. 4](#_Toc1249357)

[4. Руководство программиста. 5](#_Toc1249358)

[4.1. Описание структуры программы. 5](#_Toc1249359)

[4.2. Описание структур данных. 5](#_Toc1249360)

[4.3. Описание алгоритмов. 6](#_Toc1249361)

[5. Заключение. 7](#_Toc1249370)

[6. Литература. 8](#_Toc1249371)

# Введение.

**Стек** — [абстрактный тип данных](https://ru.wikipedia.org/wiki/%D0%90%D0%B1%D1%81%D1%82%D1%80%D0%B0%D0%BA%D1%82%D0%BD%D1%8B%D0%B9_%D1%82%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), представляющий собой [список элементов](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), организованных по принципу FILO – первым вошел, последним вышел. Чаще всего принцип работы стека сравнивают со стопкой тарелок: чтобы взять вторую сверху, нужно снять верхнюю.

Особенностью стека на списках является то, что каждый элемент стека указывает на следующий элемент.

Из стека нельзя извлечь произвольный элемент. Есть только вершина стека, и мы можем или положить туда элемент, или забрать его оттуда. На элемент в вершине стека не указывает никакой другой элемент стека.

# Постановка задачи.

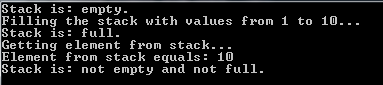
В данной лабораторной работе нужно разработать эффективную библиотеку для хранения и работы с такой структурой данных, как стек на списках.

Для этого нам нужно:

* Описать и реализовать класс **TStackList**.
* Протестировать класс **TStackList** с помощью Google Test.
* Реализовать класс **TMyException** для обработки исключений, которые могут возникнуть в результате работы класса **TStackList**.
* Продемонстрировать работу класса **TStackList**.

# Руководство пользователя.

Пример работы класса **TStackList**:



*Рис 1. Пример работы программы*

Программа работает следующим образом:

* Создается стек на списках (в данном примере его размер равен 10).
* Проверяем стек на пустоту.
* Заполняем стек значениями от 1 до 10.
* Проверяем стек на полноту.
* Забираем элемент с вершины стека.
* Проверяем стек на полноту и пустоту.

# Руководство программиста.

# Описание структуры программы.

Программа состоит из модулей:

* **StackList** – содержит в себе файл **stacklist\_main.cpp** с реализацией примера использования класса **TStackList.**
* **StackListLib –** содержит в себе файл **StackList.h**, в котором описан и реализован класс **TStackList**.
* **StackListTest** – содержит в себе файл **test\_stacklist.cpp**, в котором находится набор тестов, для проверки работоспособности класса **TStackList**.
* **MyExceptionLib** –содержитв себе файл **MyException.h** с реализацией класса исключений **TMyException**.

# Описание структур данных.

**Класс TStackList.**

Класс **TStackList** является шаблонным наследником от класса **TList** и содержит поле со спецификатором **protected**:

* **int len** – максимальный размер стека..

Далее идут методы класса со спецификатором доступа **public**:

* **TStackList(int \_len = 5)** – конструктор-инициализатор.
* **TStackList(TStackList<T> &obj)** – конструктор копирования.
* **void Put(T elem)** – метод, помещающий элемент в вершину стека.
* **T Get()** - метод, возвращающий элемент из вершину стека.
* **int GetSize()** – метод, возвращающий текущий размер стека.
* **int GetMaxSize()** - метод, возвращающий максимальный размер стека.
* **bool IsFull()** – метод, проверяющий стек на полноту.
* **bool IsEmpty()** - метод, проверяющий стек на пустоту.

**Класс TMyException.**

Класс **TMyException** содержит одно поле со спецификатором доступа **private**:

* **string str** – строка, хранящая сообщение об ошибке.

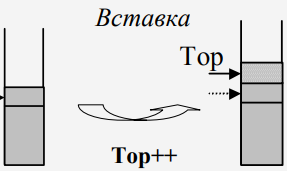
Далее идут методы класса со спецификатором доступа **public**:

* **TMyException(std::string \_str)** – конструктор инициализатор.
* **void What()** – метод вывода ошибки на экран.

# Описание алгоритмов.

**Добавление элемента в стек.**

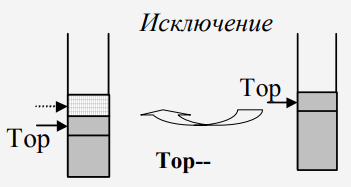
Добавляя элемент в стек, мы помещаем его в вершину стека, которая в свою очередь является первым элементом списка. После добавления элемента в стек, значение поля **top** увеличивается на 1.



*Рис. 2 Добавление элемента в стек.*

**Удаление элемента из стека.**

Удаляя элемент из стека, мы забираем его из первой непустой ячейки, которая является первым элементом в списке. На эту непустую ячейку указывает поле **top** со значением, уменьшенным на 1.



*Рис. 3 Удаление элемента из стека.*

# Заключение.

В данной лабораторной работе мне удалось реализовать библиотеку для хранения и работы со стеком на списках, а именно:

* Удалось реализовать класс **TStackList**.
* Удалось протестировать методы класса **TStackList**, а также обеспечить их работоспособность.
* Удалось реализовать класс для обработки исключений **TMyException**.

Таким образом, я смог познакомиться и реализовать структуру данных под названием стек на списках.

# Литература.

1. Ссылка из Википедии про стек:

<https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B5%D0%BA>

1. Гергель В.П. Методические материалы по курсу «Методы программирования 2», 2015.