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 массиве»**

**Выполнил:** студент группы 381706-1

Шашкин Евгений Вадимович

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

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

Нижний Новгород

2018.

Оглавление

[1. Введение. 3](#_Toc532936173)

[2. Постановка задачи. 4](#_Toc532936174)

[3. Руководство пользователя. 5](#_Toc532936175)

[4. Руководство программиста. 6](#_Toc532936176)

[4.1. Описание структуры программы. 6](#_Toc532936177)

[4.2. Описание структур данных. 6](#_Toc532936178)

[4.3. Описание алгоритмов. 7](#_Toc532936179)

[5. Заключение. 8](#_Toc532936192)

[6. Литература. 9](#_Toc532936193)

# 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.

Из стека нельзя извлечь произвольный элемент. Есть только вершина стека, и мы можем или положить туда элемент, или забрать его оттуда.

# Постановка задачи.

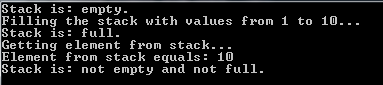
В данной лабораторной работе нужно разработать эффективную библиотеку для хранения и работы с такой структурой данных, как стек.

Для этого нам нужно:

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

# Руководство пользователя.

Пример работы класса **TStack**:



*Рис 1. Пример работы программы*

Программа работает следующим образом:

* Создается стек (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:

* **Stack** – содержит в себе файл **stack\_main.cpp** с реализацией примера использования класса **TStack.**
* **StackLib –** содержит в себе файл **Stack.h**, в котором описан и реализован класс **TStack**.
* **StackTest** – содержит в себе файл **test\_stack.cpp**, в котором находится набор тестов, для проверки работоспособности класса **TStack**.
* **MyExceptionLib** –содержитв себе файл **MyException.h** с реализацией класса исключений **TMyException**.

# Описание структур данных.

**Класс TStack.**

Класс **TStack** является шаблонным и содержит два поля со спецификатором **protected**:

* **int top** – вершина стека.
* **int size** – размер стека.
* **T\* mas** – указатель на область памяти для хранения стека.

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

* **TStack(int \_size=0)** – конструктор инициализатор.
* **TStack(TStack<T> &obj)** – конструктор копирования.
* **~TStack()** – деструктор.
* **void Put(T elem)** – функция, помещающая элемент в вершину стека.
* **T Get()** – функция, возвращающая элемент с вершины стека.
* **bool IsEmpty()** – функция, проверяющая стек на пустоту.
* **bool IsFull()** – функция, проверяющая стек на полноту.

**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**:

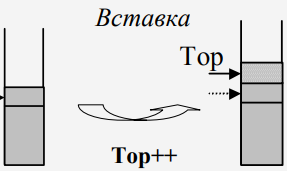
* **TMyException(std::string \_str)** – конструктор инициализатор.
* **void what()** – метод вывода ошибки на экран.

# Описание алгоритмов.



**Добавление элемента в стек.**

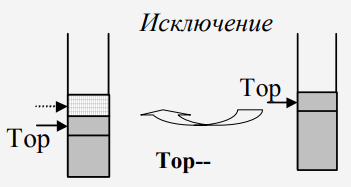
Добавляя элемент в стек, мы помещаем его в вершину стека, на которую указывает поле **top**. После добавления элемента в стек, значение поля **top** увеличивается на 1.



*Рис. 2 Добавление элемента в стек.*

**Удаление элемента из стека.**

Удаляя элемент из стека, мы забираем его из первой непустой ячейки. 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, а именно:

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

Таким образом, я смог реализовать структуру данных под названием стек.

# 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.