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**

**«Реализация стека на основе одномерного**

**динамического массива»**

**Выполнил:** студент группы 381706-1

Соболева Ю.А.

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

аспирант каф. МОСТ ИИТММ

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

Нижний Новгород

2018.

Содержание

[1. Введение 3](#_Toc533027637)

[2. Постановка задачи 4](#_Toc533027638)

[3. Руководство пользователя 5](#_Toc533027639)

[4. Руководство программиста 6](#_Toc533027640)

4.[1. Описание структуры программы 6](#_Toc533027641)

4.[2. Описание структур данных 6](#_Toc533027642)

4.[3. Описание алгоритмов 7](#_Toc533027643)

[5. Заключение 8](#_Toc533027644)

[6. Литература 9](#_Toc533027645)

# **Введение**

**Структура данных** - программная единица, которая определяет метод хранения и обработки различных логически связанных данных в вычислительной технике. Знание структур данных позволяет наиболее компактно и практично расположить данные в памяти компьютера. В данной работе мы рассмотрим такую структуру данных, как **стэк**.

**Стек** – это последовательный список с переменной длиной, в котором включение и исключение элементов происходит только с одной стороны – с его вершины. Стек функционирует по принципу: последним пришел – первым ушел, Last In – First Out (LIFO).

Программная реализация стека возможна на основе различных структур данных, например, с использованием статических или динамических одномерных массивов (векторов) и линейных списков.

**Целью данной лабораторной работы** является разработка структуры данных для хранения стеков с использование динамических массивов, а также освоение таких инструментов разработки программного обеспечения, как система контроля версий Git и фрэймворк для разработки автоматических тестов Google Test.

# **Постановка задачи**

Выполнение работы предполагает решение следующих задач:

1. Реализация класса стэка TStack.
2. Разработка интерфейса для данных классов.
3. Обеспечение работоспособности примера использования.
4. Реализация нескольких тестов на базе Google Test.

# **Руководство пользователя**

Пример использования стека:

* При запуске программы пользователю предлагается ввести число элементов в стеке, затем создается стек заданного размера с элементами типа int;
* Полученный стек выводится на экран;
* Далее требуется ввести элементы стека;
* Извлекается один элемент и стек снова выводится на экран.

# **Руководство программиста**

## **Описание структуры программы**

**Проект “Stack” состоит из следующих файлов:**

1. mainStack.cpp (в нём находится main)

**Проект “ Stack Lib ” состоит из следующих файлов:**

1. Stack.h (описание класса “TStack” реализация методов класса “TStack” и перегрузка операций )
2. StackLib.cpp

**Проект “ Test” состоит из следующих файлов:**

1. test\_stacklib.cpp (реализация тестов для класса TStack)

## **Описание структур данных**

***Класс TStack:***

*Поля:*

* int size; - размер стека;
* int top; - элемент, расположенный на вершине стека;
* T\* mas; - массив элементов стека.

*Конструкторы и деструктор:*

* TStack(int n = 0); - конструктор инициализации;
* TStack(TStack<T> &st); - конструктор копирования;

*Методы:*

* T Get() ;- забирает элемент из стека (если стек не пуст);
* void Put(T а); - добавляет элемент в стек (если стек не полон);
* bool IsFull(); - проверяет полон ли стек;
* bool IsEmpty(); - проверяет пуст ли стек;

*Перегруженные операторы:*

Нет перегруженных операторов.

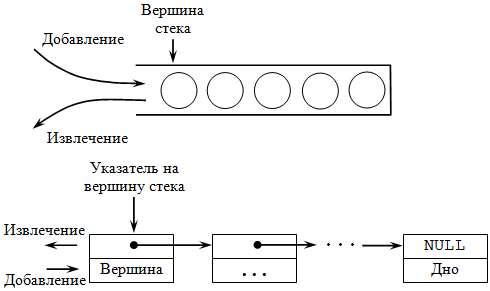
## **Описание алгоритмов**

Метод **Put:**

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.

Метод **Get:**

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



# **Заключение**

В ходе выполнения лабораторной работы я смогла реализовать такую структуру данных, как стэк. Вместе с ней, согласно заданному интерфейсу, я реализовала класс TStack. Написание нескольких своих тестов, помогло мне разобраться с системой автоматических тестов Google Test.

В результате проделанной работы у меня получилось

1. Реализовать класс стэка TStack.

2. Разработать интерфейс для данных классов.

3. Обеспечить работоспособность примера использования.

4. Реализовать некоторые тесты на базе Google Test.

Таким образом, данная лабораторная работа отвечает всем поставленным задачам. Это означает, что цель работы была достигнута.

# **Литература**

1. Лабораторный практикум. Составители: Барышева И.В., Мееров И.Б., Сысоев А.В., Шестакова Н.В. Под редакцией Гергеля В.П. Учебно-методическое пособие. – Нижний Новгород: Нижегородский госуниверситет, 2017. – 105с.

URL: <http://www.unn.ru/books/met_files/Pract_ADS.pdf>

1. Национальный открытый университет «Интуит». Курс «Основы программирования». Лекция 11: Структуры данных: общее понятие, реализация. Простейшие структуры данных: очередь, стек. Использование стека и обратная польская запись.

URL: <https://www.intuit.ru/studies/courses/2193/67/lecture/1980?page=3>