МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Национальный исследовательский**

**Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**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

Колесова Кристина Юрьевна

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

ассистент каф. МОСТ ИИТММ

Лебедев Илья Геннадьевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись

Нижний Новгород

2018.

Оглавление

[1. Введение 3](#_Toc534392616)

[2. Постановка задачи 4](#_Toc534392617)

[3. Руководство пользователя 5](#_Toc534392618)

[4. Руководство программиста 6](#_Toc534392619)

[4.1. Описание структуры программы 6](#_Toc534392620)

[4.2. Описание структур данных 6](#_Toc534392621)

[4.3. Описание алгоритмов 6](#_Toc534392622)

[5. Заключение 9](#_Toc534392623)

[6. Список литературы 10](#_Toc534392624)

# Введение

Стек на списках — структура данных, представляющая собой упорядоченный набор элементов одного типа, фиксированного количества, связанных между собой последовательно посредством указателей, организованных по принципу LIFO (англ. last in — first out, «последним пришёл — первым вышел»). Каждый элемент стека на списках имеет указатель на следующий элемент. Последний элемент стека на списках указывает на NULL. Элемент, на который нет указателя, является первым (головным) элементом списка. В стек на списках все новые элементы добавляются в начало. Вершиной стека на списках является первый элемент в списке.

Цель данной лабораторной работы – разработка структуры хранения стека на списках.

# Постановка задачи

Выполнение работы предполагает решение следующих задач:

1. Разработка и реализация класса стека на списках – TStackList
2. Реализация класса для обработки исключений – EexceptionLib
3. Реализация тестов на базе TStackList.
4. Пример использования класса TStackList.

# Руководство пользователя

При запуске программы пользователь сможет увидеть простейший пример работы со стеком на списках.

Пользователя просят ввести число n, затем создается стек размера n. Далее необходимо ввести элементы стека. После того, как все элементы будут введены, полученный стек выводится на экран. Затем из стека удаляется элемент, полученный в результате этого стек также выводится на экран.

# Руководство программиста

## Описание структуры программы

В программе содержатся следующие модули:

* Модуль StackListLib – статическая библиотека, содержащая файлы StackList.h и StackList.cpp, в которых описан и реализован класс TStackList.
* Модуль ExceptionLib – библиотека, позволяющая создавать собственные исключения.
* Модуль ListLib – статическая библиотека, содержащая файлы TList.h, TList.cpp, TElement.h и TElement.cpp, в которых описаны и реализованы классы TElement и TList
* StackList– содержит файл StackList.cpp, в котором реализован простой пример использования стека
* Модуль StackListTest – содержит файл test.cpp с тестами для класса TStackUsingList

## Описание структур данных

* Класс TStackList

Является шаблонным классом, наследуется от TList.

Элементы класса, объявленные со спецификатором protected:

int st\_size – максимальный размер стека на списках

Элементы класса, объявленные со спецификатором public:

TStackUsingList <T>(int \_size = 5) - конструктор;

TStackUsingList <T>(TStackUsingList<T> &A) – конструктор копирования; ~TStackUsingList() - деструктор;

T Get() – получить значение с вершины стека;

void Del() – удалить значение с вершины стека;

void Put(T A) – положить значение на вершину стека;

void Show() – вывод содержимого стека на экран;

int GetSize() – получить размер (количество элементов);

int GetMSize() – получить максимальный размер;

bool IsFull() – проверка на полноту;

bool IsEmpty() – проверка на пустоту

## Описание алгоритмов

* Добавление и изъятие элемента в стеке на списках

При добавлении элемента в стек, он размещается на вершине стека. При изъятии элемента из стека берется элемент, находящийся на вершине стека. Поскольку в данной работе стеки хранится на списках, то целесообразно вершиной стека считать первый элемент списка. То есть добавление и изъятие элементов происходит только из начала списка. В этом случае сложность этих операций составит О(1). Иначе, для добавления или изъятия элементов из конца списка, пришлось бы обходить все список поэлементно, и сложность операций составила O(n).

# Заключение

В результате выполнения лабораторной работы была разработана библиотека, реализующая шаблонный класс стека на списках TStackList. В ней реализованы методы работы со стеком, описанные в разделе «Структуры данных».

Были реализованы тесты для проверки работоспособности класса TStackList на базе GoogleTest. А также приведен пример работы класса.

# Список литературы

1. Википедия. Статья «Стек»: [https://ru.wikipedia.org/wiki/Стек].
2. Васильев А.Н. Самоучитель С++ с примерами и задачами. -СПб.: Наука и Техника, 2016. -480с.
3. Гергель В.П. Методические материалы по курсу «Методы программирования 2», Нижний Новгород, 2015.