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 ЛАБОРАТОРНОЙ РАБОТЕ №9**

**«Стек на списке»**

**Выполнил:** студент группы 381903-3

Зинкин Кирилл Сергеевич

**Проверил:**

ассистент каф. МОСТ ИИТММ Лебедев Илья Геннадьевич

Нижний Новгород

2020.

Содержание

[1. Введение 3](#_Toc536523907)

[2. Постановка задачи 4](#_Toc536523908)

[3. Руководство пользователя 5](#_Toc536523909)

[4. Руководство программиста 6](#_Toc536523910)

[4.1. Описание структуры программы 6](#_Toc536523911)

[4.2. Описание структур данных 6](#_Toc536523912)

[4.3. Описание алгоритмов 7](#_Toc536523913)

[5. Заключение 8](#_Toc536523914)

[6. Литература 9](#_Toc536523915)

# 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](https://ru.wikipedia.org/wiki/NULL_(%D0%A1%D0%B8)). 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 списке.

# Постановка задачи

Выполнение работы предполагает решение следующих задач:

1. Разработка и реализация класса стека на списках – TStackList.
2. Создание класса для обработки исключений – Exception, которые могут возникнуть при выполнении различных операций.
3. Разработка программы, демонстрирующей работу классa TStackList.
4. Реализация набор автоматических тестов с использованием гугл тестов.

# Руководство пользователя

Рассмотрим пример использования класса TStackList.

При запуске программы c пользователя спрашивается максимальный размер стека . Затем создается стек размером n и автоматически заполняется числами от 1 до n. Добавление каждого числа в стек сопровождается соответствующим сообщением на экране. После заполнения полученный стек выводится на экран. Далее происходит изъятие элементов из стека до полного его опустошения. Изъятие каждого элемента также сопровождается соответствующим сообщением на консоли. После того, как будут извлечены все элементы, на экран поступит сообщение о пустоте стека. На этом работа программы прекращается.

# Руководство программиста

## Описание структуры программы

Программа состоит из следующих модулей:

* Модуль StackList. Содержит пример использования стека. Реализация в файле *main\_StackList.cpp.*
* Модуль StackListLib – статическая библиотека. Содержит файл StackList.h, в котором описан интерфейс и реализация шаблонного класса *TStackList* (2 конструктора, деструктор и 7 методов).
* Модуль StackListTest. Содержит 14 тестов, описанных в файле *StackTest.cpp* и разработанных с помощью использования Google C++ Testing Framework.
* Модуль ExceptionLib – библиотека, содержащая класс исключений.

## Описание структур данных

#### Класс TStackList

Класс *TStackList* является шаблонным классом. Наследуется от класса *TList* со спецификатором public. В классе *TStackList* определено одно поле, со спецификатором доступа protected, int size – максимальный размер стека.

Далее в публичной зоне (public):

1. Прописаны 2 конструктора и деструктор:

* *TStackList(int \_size = 10)* – конструктор с одним параметром.
* *TStackList(TStackList<T> &A)* – конструктор копирования.
* *~TStackList()* – деструктор.

1. Прописаны методы для работы со стеком на списках:

* *void Put(T A)* – добавить новый элемент A в начало списка.
* *T Get()* – изъять с удалением элемент находящийся в начале списка.
* *void Print()* – вывод элементов стека на списках на экран.
* *bool IsFull()* – проверка стека на списках на полноту.
* *bool IsEmpty()* – проверка стека на списках на пустоту.
* *int GetSize()* – получить текущее количество элементов в стеке на списках.
* *int GetMaxSize()* – получить размер стека.

## 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 ходе выполнения лабораторной была изучена структура данных – стек на списке. Была разработана библиотека StackListLib, реализующая шаблонный класс стека TStackList. В ней реализованы методы работы со стеком описанные в разделе «Структуры данных».

Программная реализация стека на массиве была продемонстрирована на примере, описывающем основные методы класса TStackList.

# Литература

* Книги
  + - 1. A.O. Грудзинский. Методы программирования, Издательство Нижегородского госуниверситета, 2006.
      2. Васильев А.Н. Самоучитель С++ с примерами и задачами. -СПб.: Наука и Техника, 2016. -480с.
* Ссылки в Internet
  + - 1. Гергель В.П. Методические материалы по курсу «Методы программирования 2»: [http://www.itmm.unn.ru/files/2018/11/Primer-1.6.-Struktura-hraneniya-neskolkih-stekov-s-ispolzovaniem-spiskov.pdf], 2015.
      2. Википедия: свободная электронная энциклопедия: на русском языке: <https://ru.wikipedia.org/wiki/Стек>
      3. Википедия: свободная электронная энциклопедия: на русском языке: <https://ru.wikipedia.org/wiki/Связный> список