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

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)

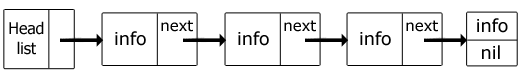
# **Введение**

**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 на** **списках**.

**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.

**Линейный список –** это структура данных, представляющая собой конечное множество упорядоченных элементов, связанных друг с другом посредствам указателей. Каждый элемент структуры содержит поле с какой-либо информацией, а также указатель на следующий элемент.



**Целью данной лабораторной работы** является разработка структуры данных для хранения стэков на основе связных линейных списков, 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 списка TStackList, унаследованного от класса TList.
2. Разработка интерфейса для данных классов.
3. Обеспечение работоспособности примера использования.
4. Реализация нескольких тестов на базе Google Test.

# **Руководство пользователя**

При запуске программы на экран выводится сообщение, что все тесты пройдены успешно. Примеры вывода сообщений для класса TStackList показаны на рис.1

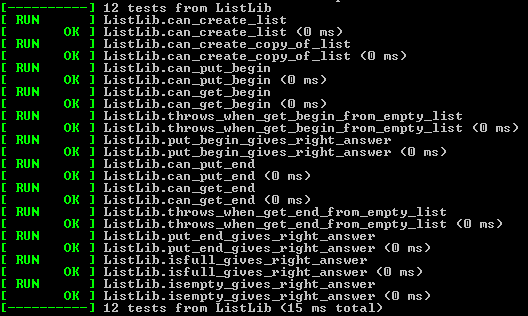


Рис. 1

В качестве примера использования я привела проверку стэка на пустоту и вывод всего стэка . То есть, в самой программе мы создали стэк на списке и положили туда 3 элемента:21, 29 и 213.

Результат работы этой программы показан на рис 2.



Рис. 2

# **Руководство программиста**

## **Описание структуры программы**

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

1. mainStackList.cpp (в нём находится main)

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

1. StckListLib.h (описание класса “TStackList” реализация методов класса “TStackListList” и перегрузка операций )
2. StackListLib.cpp

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

1. test\_stacklistlib.cpp (реализация тестов для класса TStackList)

## **Описание структур данных**

***Класс TStackList:***

*Поля:*

* TElem<T>\* begin; -указатель на начало списка;

*Конструкторы и деструктор:*

* TStackList<T> (int \_size = 10); - конструктор инициализации;

TStackList<T> (TStackList<T> &A); - конструктор копирования;

~TStackList<T> (); - деструктор;

*Методы:*

* void Put(T A); - добавляет элемент в стэк;
* T Get(); - забирает элемент из стэка;
* int GetSize(); - возвращает размер стэка;
* int GetCount(); - возвращает количество занятых ячеек памяти;
* bool IsEmpty(); - проверяет пуст ли стэк;
* bool IsFull(); - проверяет полон ли стэк;

*Перегруженные операторы:*

Нет перегруженных операторов.

## **Описание алгоритмов**

**Метод Put:**

При добавлении элемента в стек, он размещается на вершине стека. Поскольку в данной работе стеки хранится на списках, то целесообразно вершиной стека считать первый элемент списка. То есть добавление происходит лишь к началу списка. Иначе, для добавления элементов в конец списка, пришлось бы обходить все список поэлементно.

**Метод Get:**

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 изъятие элементов происходит только из начала списка. Иначе, для изъятия элементов из конца списка, пришлось бы обходить все список поэлементно.

# **Заключение**

В ходе выполнения лабораторной работы я смогла реализовать такую структуру данных, как список на указателях. Вместе с ней, согласно заданному интерфейсу, я реализовала класс TStackList, унаследованный от класса TList. Написание нескольких своих тестов, помогло мне разобраться с системой автоматических тестов Google Test.

В результате проделанной работы у меня получилось

1. Реализовать класс стэка на списках TList.

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. URL: http://kvodo.ru/data-structures-introduction.html
2. A.O. Грудзинский. Методы программирования, Издательство Нижегородского госуниверситета, 2006.