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 данных – стек на списках»**

**Выполнил:** студент группы 381706-1

Голованова Елена Александровна

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

ассистент каф. МОСТ ИИТММ

Лебедев Илья Геннадьевич

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

Нижний Новгород

2018.

Оглавление

[1. Введение 3](#_Toc534459553)

[2. Постановка задачи 4](#_Toc534459554)

[3. Руководство пользователя 5](#_Toc534459555)

[4. Руководство программиста 6](#_Toc534459556)

[a. Описание структуры программы 6](#_Toc534459557)

[b. Описание структур данных 6](#_Toc534459558)

[c. Описание алгоритмов 7](#_Toc534459559)

[5. Заключение 9](#_Toc534459560)

[6. Список литературы 10](#_Toc534459561)

# Введение

В мире существуют множество разных структур данных. Они созданы для эффективного использования памяти компьютера, так как вместимость – исчерпывающая величина: ее никогда не хватает.

Бывают разные структуры данных. Например, множества, очереди, maps. В этой лабораторной работе мы рассмотрим такую структуру, как стек на списках.

**Стек на списках** — такая структура данных, которая представляет собой набор идущих по порядку элементов одного типа, определенного количества, связанных между собой с помощью указателей. Стек на списках организован по принципу LIFO (англ. last in — first out, «последним пришёл — первым вышел»). У каждого элемента данной структуры присутствует указатель на следующий элемент. Пырвым элементом считается тот, у которого нет указателя. Последний элемент стека на списках указывает на [NULL](https://ru.wikipedia.org/wiki/NULL_(%D0%A1%D0%B8)). Особенностью стека на списках является то, что все новые элементы добавляются в начало.  Вершиной стека на списках является первый элемент в списке.

Цель данной лабораторной работы – реализовать такую структуру данных как стек на списках.

# Постановка задачи

Выполнение работы предполагает решение следующих задач:

1. Реализация класса TStackList.
2. Выполнение определенных функций, операторов и конструкторов в ранее упомянутых классах.
3. Создать класс TException во избежания ошибок.
4. Выполнение примера использования программы.
5. Реализация простых тестов на базе Google Test.

# Руководство пользователя

Работа пользователя происходит так:

1. В начале программы у пользователя спрашивают максимальное количество элементов, то есть размер стека на списках.
2. После этого автоматически создается стек. Он заполняется числами от одного до числа, равного количеству элементов.
3. После этого на экране выводится полученный стек.
4. Следующим действием будет изъятие элементов.
5. Последним пунктом программы является вывод сообщения о том, что стек на списках пуст.

# 4. Руководство программиста

## Описание структуры программы

Для реализации лабораторной работы создается несколько модулей:

* Модуль List

Этот модуль содержит реализацию примера использования программы.

* Модуль ListLib

Этот модуль содержит реализацию списка. Содержит файл List.h, в котором описан интерфейс и реализация шаблонного класса TList. А также содержит файл Elem.h, в котором представлен интерфейс и реализация вспомогательного шаблонного класса TElem.

* Модуль ListTest

Этот модуль содержит тесты, проверяющие работу каждого методов класса TList. Они разработаны с помощью использования Google C++ Testing Framework.

* Модуль StackList

Этот модуль содержит реализацию примера использования программы.

* Модуль StackListLib

Этот модуль содержит реализацию стека на списках. Содержит файл StackList.h, в котором описан интерфейс и реализация шаблонного класса TStackList.

* Модуль StackListTest

Этот модуль содержит тесты, проверяющие работу каждого методов класса TStackList. Они разработаны с помощью использования Google C++ Testing Framework.

* Модуль Exception

Этот модуль содержит класс исключений.

## Описание структур данных

Класс TElem – класс узлов списка. Он является шаблонным.

1. В зоне protected определены два поля:

* T dannyye - переменная под хранение данных;
* TElem<T>\* next - указатель на следующий узел.

2. В зоне public:

* TElem<T>(T \_dannyye = 0, TElem<T>\* n = 0) - конструктор ;
* Elem<T>(TElem<T>& A) - конструктор копирования;
* T GetDannyye() - получение значения из узла;
* TElem<T>\* GetNext() - получение указателя на следующий узел;
* void SetDannyye(T \_dannyye) - установление данных;
* void SetNext(TElem<T>\* n) - установления указателя на следующее звено списка.

Класс TList – класс шаблонный.

1. В зоне protected определены два поля:

* TElem<T>\* begin - указатель на элемент в начале списка;
* int num - количество элементов в списке.

2. В зоне public:

* TList() - конструктор по умолчанию;
* TList(TList<T>& F) - конструктор копирования;
* virtual ~TList() – деструктор;
* int GetSize() - получение количества элементов в листе;
* void Put(int \_n, T elem) - установление элемента на позицию n в списке;
* T Get(int \_n) - получение элемента из списка;
* void PutBegin(T A) – установление элемента в начало;
* void PutEnd(T A) – установление элемента в конец;
* T GetBegin() - получение элемента в начале списка;
* T GetEnd() - получение элемента в конце списка;
* bool IsFull() - проверка на полноту;
* bool IsEmpty() - проверка на пустоту;
* void Print() - печать листа.

Класс TStackList – класс шаблонный. Наследуется от класса TListсо спецификатором public.

1. В зоне protected определено одно поле:

* int size – максимальный размер стека.

2. В зоне public:

* TStackList() - конструктор по умолчанию с параметром;
* TStackList(TStackList<T>& А) - конструктор копирования;
* ~TStackList() – деструктор;
* void Put(T А) – добавление нового элемента в начало;
* T Get () – изъятие с удалением;
* int GetMaxSize() - получение размера стека;
* int GetSize() - получение текущего количества элементов;
* bool IsEmpty() - проверка на пустоту;
* bool IsFull() - проверка на полноту;
* void Print() - печать стека на списках.

## Описание алгоритмов

* Добавление и изъятие элемента в стек на списках:

Эти операции могут происходить только таким образом. Элемент может добавляться только на его вершину. Изыматься может элемент, находящийся на вершине стека. Такой порядок дает быстроту действий и маленькую сложность.

# 5. Заключение

В результате выполнения данной лабораторной работы была разработана структура хранения данных – стек на списках. Они были реализованы при помощи нескольких классов (их реализация была описана выше).

Помимо реализации в лабораторной работе есть пример использования.

Получены новые знания и навыки.

Были реализованы тесты для проверки работоспособности вышеперечисленных классов на базе GoogleTest.

# 6. Список литературы

1. Википедия. Статья «Структура данных»

[https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0\_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85]

1. Васильев А.Н. Самоучитель С++ с примерами и задачами. -СПб.: Наука и Техника, 2016. -480с.
2. Гергель В.П. Методические материалы по курсу «Методы программирования 2», Нижний Новгород, 2015.
3. Пост на habr.ru

[https://m.habr.com/ru/post/334914/]