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**

**«Список»**

**Выполнил:** студент группы 381706-1

Корнев Никита Алексеевич

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

ассистент каф. МОСТ ИИТММ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лебедев И.Г

Нижний Новгород

2018.

Содержание

[1. Введение 3](#_Toc533799739)

[2. Постановка задачи 4](#_Toc533799740)

[3. Руководство пользователя 5](#_Toc533799741)

[4. Руководство программиста 6](#_Toc533799742)

[4.1. Описание структуры программы 6](#_Toc533799743)

[4.2. Описание структур данных 7](#_Toc533799744)

[4.3 Описание алгоритмов 8](#_Toc533799745)

[5. Заключение 9](#_Toc533799746)

[6. Литература 10](#_Toc533799747)

# Введение

Структура данных (англ. data structure) — программная единица, позволяющая хранить и обрабатывать множество однотипных и/или логически связанных данных в вычислительной технике. Для добавления, поиска, изменения и удаления данных структура данных предоставляет некоторый набор функций, составляющих её интерфейс. Одной из важных структур является список.

Список — базовая динамическая структура данных в информатике, состоящая из узлов, каждый из которых содержит как собственно данные, так и одну или две ссылки («связки») на следующий и/или предыдущий узел списка. Принципиальным преимуществом перед массивом является структурная гибкость: порядок элементов связного списка может не совпадать с порядком расположения элементов данных в памяти компьютера, а порядок обхода списка всегда явно задаётся его внутренними связями.

Линейный однонаправленный список — это структура данных, состоящая из элементов одного типа, связанных между собой последовательно посредством указателей. Каждый элемент списка имеет указатель на следующий элемент. Последний элемент списка указывает на NULL. Элемент, на который нет указателя, является первым (головным) элементом списка. Здесь ссылка в каждом узле указывает на следующий узел в списке. В односвязном списке можно передвигаться только в сторону конца списка. Узнать адрес предыдущего элемента, опираясь на содержимое текущего узла, невозможно.

В данной лабораторной работе нам предстоит написать собственную реализацию данной структуры.

# Постановка задачи

1. Написать класс, реализующий список.
2. Написать тесты на основе Google Tests для проверки работы класса.

# Руководство пользователя

Чтобы использовать данный класс в своем проекте, необходимо подключить библиотеку «List.h». Библиотека позволяет:

* Создавать объекты типа *список*:

List li1; **конструктор по умолчанию**

List li(10); **конструктор с параметром**

List li3 (li22); **конструктор копирования**

* Добавлять элементы в начало и конец *списка*:

li.PutBegin(T A);

li.PutEnd(T A);

* Брать элементы из начала и конца *списка*:

A = li1.GetBegin();

B = li1.GetEnd();

* Проверять на *пустоту*:

A = li1.IsEmpty();

* Проверятьна *полноту*:

A = li1.IsFull();

# Руководство программиста

## Описание структуры программы

Проект состоит из следующих модулей:

1. List

Модуль cодержит файл код «main.cpp», в котором продемонстрирован пример использования структуры.

1. ListLib

Модуль содержит файл заголовок «List.h», описывающий структуру «очередь», файл заголовок «Elem.h, описывающий структуру элементов списка, а также файл кода «main.cpp». Так как класс шаблонный, main содержит лишь подключение.

1. ListTest

Модуль содержит файлы кода «test\_main.cpp», «test\_list.cpp». В последнем реализованы тесты для проверки корректности работы методов данного класса.

## Описание структур данных

#### *Структура Elem*

Поля со спецификатором доступа «protected»:

* **T elem;** *// Значение*
* **TElem<T>\* next;** *// Указатель на след элемент*

Поля со спецификатором доступа «public»:

* **TElem(T \_e = 0, TElem<T>\* \_n = 0*)****; // Конструктор*
* **TElem(TElem<T> &A);** *// Конструктор копирования*
* **T Get();** *// Получить эначение элемента*
* **TElem\* GetNext();** *// Получить указатель на следующий*
* **void Set(T e);** *// Установить значение*
* **void SetNext(TElem<T>\* n);** *// Установить указатель*

#### Структура List

Поля со спецификатором доступа «protected»:

* **TElem<T>\* begin;** *// Указатель на первое звено списка*

Поля со спецификатором доступа «public»:

* **TList();** *// Конструктор по умолчанию*
* **TList(TList<T> &A);** *// Конструктор копирования*
* **void PutBegin(T A);** *// Установить начальное звено*
* **void PutEnd(T A);** *// Установить последнее звено*
* **T GetBegin();** *// Взять начало*
* **T GetEnd();** *// Взять конец*
* **bool IsFull();** *// Проверка на полноту*
* **bool IsEmpty();** *// Проверка на пустоту*

## Описание алгоритмов

Описание некоторых алгоритмов:

* **TList();** *создает объект типа list с полем begin равным 0*
* **TList(TList<T> &A);** *создает list со значениями полей объекта A*
* **void PutBegin(T A);** *если список пуст, создает Elem со значением A, указывающий на 0, в противном случае, создает временнный указатель temp со значением A, указывающий на прежний элемент, являющийся началом, после чего присваивает значение temp началу списка*
* **void PutEnd(T A);** *Аналоигчно PutBegin*
* **T GetBegin();** *Возвращает значение первого элемента*
* **T GetEnd();** *Возвращает значение последнего элемента*
* **bool IsEmpty();** *Если начало равно нулю, возвращает единицу, в противном случае 0*

# Заключение

В данном лабораторной работе мне удалось:

* Успешно реализовать класс для списка
* Продемонстрировать пример использования данного класса
* Написать тесты на основе Google Tests для проверки корректной работы данного класса

# Литература

Интернет-источники:

1. <https://ru.wikipedia.org/wiki/Структура_данных>
2. [https://ru.wikipedia.org/wiki/Список\_(информатика)](https://ru.wikipedia.org/wiki/Очередь_(программирование))
3. <https://habr.com/ru/post/232009/>

Книги:

1. Гергель В.П. Методические материалы по курсу «Методы программирования 2», Нижний Новгород, 2015.
2. A.O. Грудзинский. Методы программирования, Издательство Нижегородского госуниверситета, 2006.