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**

**ЛАБОРАТОРНАЯ РАБОТА №6**

**«Список на указателях»**

**Выполнил:** студент группы 381706-2

Макарихин Семён Андреевич

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Руководитель:**

Ассистент кафедры МОСТ

Лебедев Илья Геннадьевич

\_\_\_\_\_\_\_\_\_\_\_ Подпись

Нижний Новгород

2019

Содержание

[1.Введение 3](#_Toc14752)

[2. Цели и задачи 5](#_Toc14753)

[3. Руководство пользователя 5](#_Toc14755)

[4. Руководство программиста 6](#_Toc14756)

[4.1. Описание структуры программы 6](#_Toc14757)

[4.2. Описание функций и процедур, их алгоритмов 6](#_Toc14758)

[5. Эксперименты](#_Toc14759) 9

[6. Заключение 1](#_Toc14760)0

[7. Литература 1](#_Toc14761)1

[8. Приложение 1](#_Toc14762)2

# 1.Введение

Списки являются чрезвычайно гибкой структурой, так как их легко сделать большими или меньшими, и их элементы доступны для вставки или удаления в любой позиции списка. Списки также можно объединять или разбивать на меньшие списки. Списки регулярно используются в приложениях, например в программах информационного поиска, трансляторах программных языков или при моделировании различных процессов.

В математике список представляет собой последовательность элементов определенного типа ( elementtype ). Представим список в виде последовательности элементов, разделенных запятыми: a1, а2, ..., аn, где n≥ 0 и всё ai имеют тип elementtype. Количество элементов n - длина списка. Если n ≥ 1, то а1 называется первым элементом, а аn—последним элементом списка. В случае n = 0 имеем пустой список, который не содержит элементов.

Важное свойство списка заключается в том, что его элементы можно линейно упорядочить в соответствии с их позицией в списке. Говорим, что элемент ai предшествует ai+1 для і = 1, 2, ..., n — 1 и ai следует за ai-1 для і =2, 3, .., n. Также будем говорить, что элемент аi имеет позицию і.

# 2. Цели и задачи

В рамках лабораторной работы ставится задача эффективной реализации структуры данных – линейный односвязный список на указателях и выполнение основных операций над ним:

* добавления в начало элемента списка.
* добавления в конец элемента списка.
* добавления, извлечения промежуточного элемента списка.
* извлечения с удалением элемента из начала списка,
* извлечения с удалением элемента из конца списка,
* проверка списка на полноту/пустоту.

В процессе выполнения лабораторной работы требуется использовать систему контроля версий [Git](https://git-scm.com/book/ru/v2) и фрэймворк для разработки автоматических тестов [Google Test](https://github.com/google/googletest).

Выполнение работы предполагает решение следующих задач:

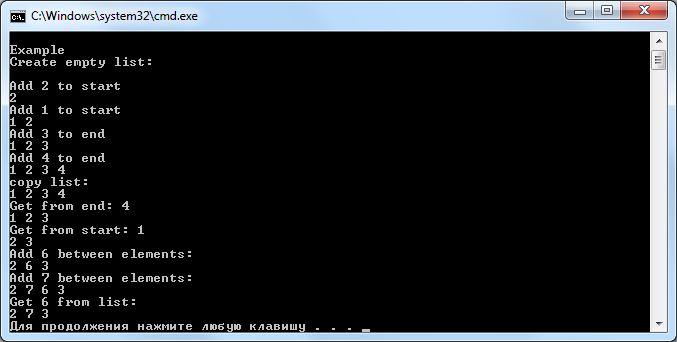
1. Реализация методов шаблонного класса TList согласно заданному интерфейсу.
2. Реализация класса для обработки исключений– TException, которые могут возникнуть при выполнении различных операций, согласно заданному интерфейсу.
3. Обеспечение работоспособности тестов и примера использования.
4. Реализация заготовок тестов, покрывающих все методы TList.
5. Модификация примера использования в тестовое приложение, позволяющее задавать списки и осуществлять основные операции над ними.

# 3. Руководство пользователя

Запускаем программу из файла sample\_tlist.cpp.

Программа выведет пример правильной работы основных операций со списком.

1. добавления в начало элемента списка.
2. добавления в конец элемента списка.
3. добавления, извлечения промежуточного элемента списка.
4. извлечения с удалением элемента из начала списка.
5. извлечения с удалением элемента из конца списка.



# 4. Руководство программиста

Разработка системы вычисления проводились в среде “Microsoft Visual Studio 2010”.

В данной работе будет использовано 3 класса:

* Класс «Элемент» (TElement) – реализует узел списка.
* Класс «Список» (TList), реализованный с использованием указателей.
* Класс исключения (TExсeption).

## 4.1. Описание структуры программы

Модульная структура программы:

1. telement.h– модуль с классом TElement, в котором определен интерфейс шаблонного класса Элемент и реализация его методов.
2. tlist.h– модуль с классом TList, в котором определен интерфейс шаблонного класса Список и реализация его методов.
3. exсeption.h – модуль с классом исключения TExсeption.
4. sample\_tlist.cpp, sample\_performance\_check.cpp– модуль программы тестирования, с которым работает пользователь, в котором проводятся эксперименты.
5. test\_main.cpp, test\_tlist.cpp, test\_telement.cpp – модуль с функциями тестирования для созданных классов. Содержит 22 теста для класса TList и 6 тестов для класса TElement.

## 4.2. Описание функций и процедур, их алгоритмов

Рассмотрим реализацию методов шаблонного класса TElement:

template <class ElementType>

class TElement

1. TElement(ElementType \_elem = 0, TElement<ElementType>\* \_next = 0) - конструктор класса с параметрами, принимающий значение элемента и указатель на следующий элемент. По умолчанию значения установятся в ноль .

2. TElement(TElement<ElementType> &Elem) - конструктор копирования.

Принимает ссылку на объект класса TElement.

3. ElementType GetElem() - метод класса, позволяющий получить значение элемента.

4. TElement\* GetNext() – метод класса, позволяющий получить указатель на следующий элемент.

5. void Set(ElementType \_elem) - метод класса, позволяющий установить значение элемента.

6. void SetNext(TElement<ElementType>\* next) - метод класса, позволяющий установить указатель.

Рассмотрим реализацию методов шаблонного класса TList:

template <class ListType>

class TList

1. TList() – конструктор класса по умолчанию.

2. TList(TList<ListType> &List) - конструктор копирования.

Принимает ссылку на объект класса TList.

3. ~TList() - деструктор. Освобождает выделенную память.

4. void Put(int \_n, ListType elem) - метод, позволяющий добавить новый элемент в список на определенную позицию.

5. void PutStart(ListType List) - метод, позволяющий добавить новый элемент в начало списка.

6. void PutEnd(ListType List) - метод, позволяющий добавить новый элемент в конец списка.

7. ListType Get(int \_n) - метод изъятия элемента на определенной позиции из списка с удалением.

8. ListType GetStart() - метод изъятия элемента из начала списка с удалением.

9. ListType GetEnd() - метод изъятия элемента с конца списка с удалением.

10. bool IsFull() - метод проверки списка на полноту.

11. bool IsEmpty() - метод проверки списка на пустоту.

12. void PrintList() - метод отображения текущих элементов списка.

# 5. Эксперименты

В качестве примера рассмотрим операцию добавления элемента в конец и в начало списка для класса TList.

Теоретическая сложность выполнения алгоритмов O(1).

Мы провели измерение скорости добавления элемента в конец и в начало списка при разном количестве элементов: 10, 100, …, 1000000. Ниже вы можете увидеть таблицу зависимости времени выполнения операции от количества элементов списка.

По приведенным данным можно сделать вывод, что практическая сложность выполнения алгоритма равна теоретической.

|  |  |  |
| --- | --- | --- |
| Кол-во элементов | метод PutEnd() | метод PutStart() |
| 10 | 0 | 0 |
| 100 | 0 | 0 |
| 1000 | 0 | 0 |
| 10000 | 0 | 0 |
| 100000 | 0 | 0 |

# 6. Заключение

В результате лабораторной работы была разработана структура данных – односвязный линейный список на указателях, а также освоены такие инструменты разработки программного обеспечения, как система контроля версий [Git](https://git-scm.com/book/ru/v2).

# 7. Литература

1. Лабораторный практикум: Учебно-методическое пособие / Мееров И.Б. [и др.] – Нижний Новгород: Нижегородский госуниверситет , 2017. – 105с.
2. <http://bookwu.net/book_algoritmy-i-struktury-dannyh_1245/12_2.2.-realizaciya-spiskov>
3. Д. Кнут. Искусство программирования. (3-е издание) Т.1.

# 8. Приложения