Федеральное агентство по образованию Российской Федерации

Государственное образовательное учреждение

Высшего профессионального образования

Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий математики механики

Выполнил:

студент ИИТММ гр. 381706-2

Антипин А.С.\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил:

ассистент каф. МОСТ, ИИТММ

Лебедев И.Г\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

Список на массивах

Отчет по лабораторной работе

Нижний Новгород

2018 г.

**Содержание**

[1.Введение 3](#_Toc1075411)

[2.Постановка целей и задач 4](#_Toc1075412)

[3.Руководство пользователя 5](#_Toc1075413)

[4.Руководство программиста 6](#_Toc1075414)

[4.1.Описание структуры программы 6](#_Toc1075415)

[4.2.Описание структур данных 7](#_Toc1075416)

[4.3.Описание алгоритмов 8](#_Toc1075417)

[5.Заключение 9](#_Toc1075418)

[6.Литература 10](#_Toc1075419)

# 1.Введение

Самым главным преимуществом списка является то, что ему не надо знать заранее сколько элементов он будет хранить. Элементы можно добавлять пока не кончится физическая память на компьютере. А что если заранее известно максимальное количество элементов, которое может храниться, такие структуры данных, как стек и очередь являются неудобными для работы с этими данными? Тогда можно попробовать создать новую структуру данных «Список на массивах», в котором известно максимальное количество элементов в списке, и присутствуют разные способы доступа к данным.

Разберем устройство этой структуры данных. Список на массивах состоит из двух массивов, один из которых – это массив элементов, которые необходимо хранить, а второй – это целочисленный массив из индексов элементов первого массива. Данный массив позволяет упорядочивать элементы первого, даже если они в разном порядке. По индексу следующего элемента можно положить или забрать элемент. Тем самым происходит работа со списком.

# 2.Постановка целей и задач

Целью лабораторной работы является создание структуры хранения данных типа «Список на массивах» и методов работы с ним, таких как:

* Добавление элементов в начало и конец списка;
* Удаление элементов из начала конца списка с их возвратом;

Для реализации алгоритмов будет использоваться шаблонный класс:

* ArrList.

Для проверки правильности работы этих классов будут написаны тесты с использованием фреймворка Google Test, а также тестовый образец программы, которая будет использует этот класс.

# 3.Руководство пользователя

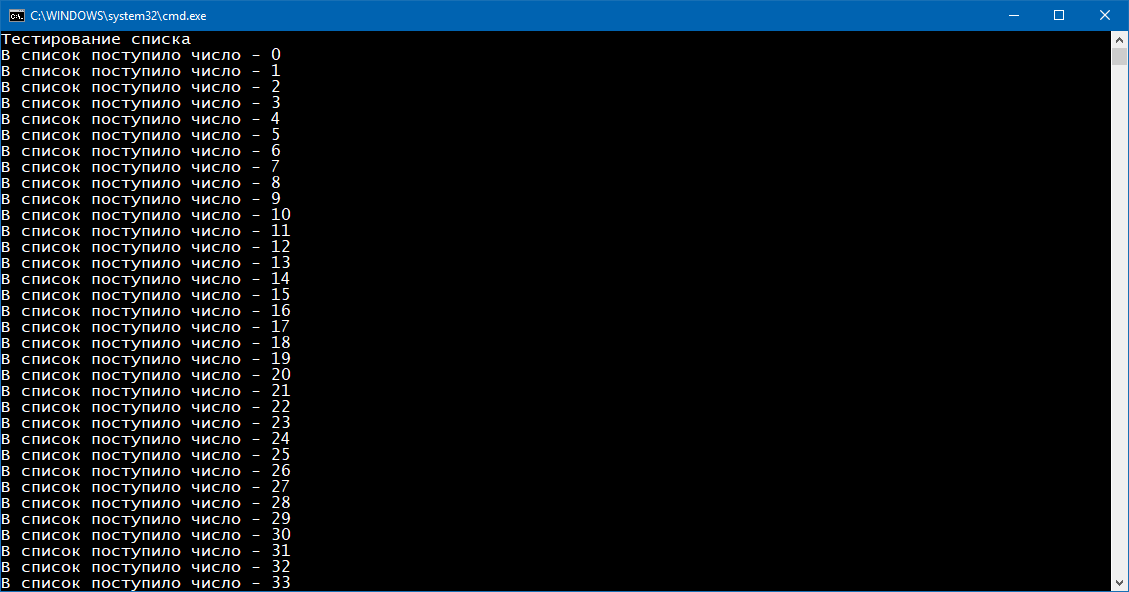
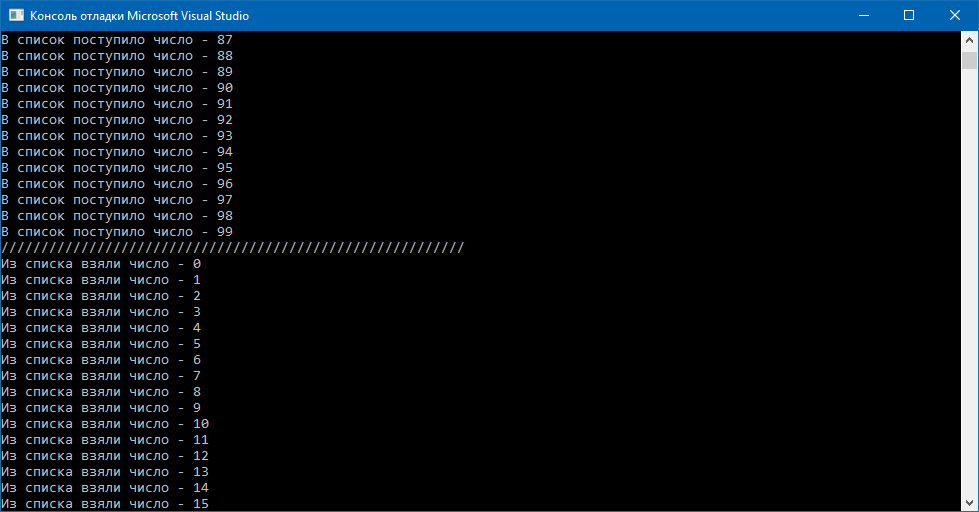
После запуска программы пользователя встречает консольное окно (рис. 1):

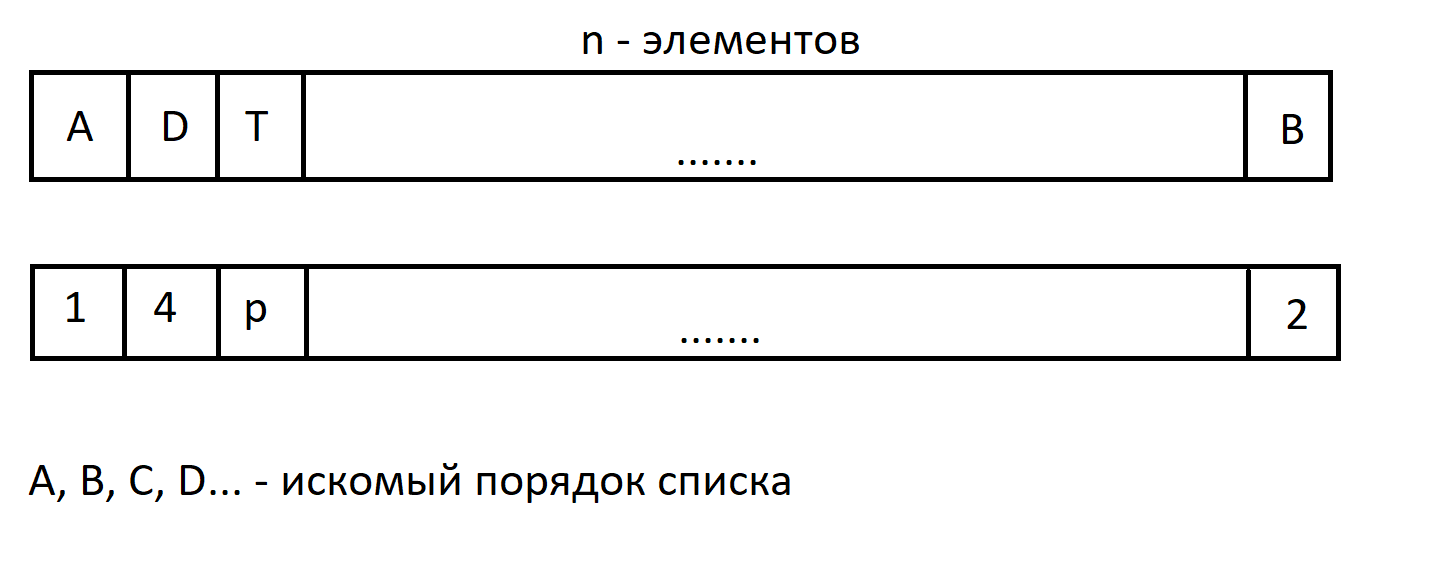
рис. 2 (возвращение чисел от 0 до 99)

рис. 1 (вывод программы тестирования списка для пользователя)

в которой сначала в список кладется числа от 0 до 99, а затем эти числа забираются из списка с удалением (рис. 2).

# 4.Руководство программиста

## 4.1.Описание структуры программы

Список будет реализован с помощью двух массивов:

Для реализации алгоритмов будет использовано класс ArrList. А также проект использующий фреймворк Google Test, для проверки правильности работы этих классов и тесовый проект, который будет показываться пользователю.

Древо классов

**Класс gtest**

**test\_main.cpp**

**Arr\_list\_test.cpp**

**Проект для пользователя**

**main.cpp**

**Arr\_List.h**

**Класс ArrList**

**Класс List:**

Класс ArrList содержит реализацию работы со списком на массивах. В нем реализованы такие методы, как положить в список, вернуть из списка элемент и др.

**Класс gtest:**

Класс gtest реализует тестирование класса ArrList, по средствам фреймворка Google Test. Тесты пишутся для каждого метода классов, каждого ветвления этих методов и для всех возможных исключений этих методов.

**Проект Array List:**

В данном проекте реализован примет использования списка, в который поочередно кладутся и забираются элементы.

## 4.2.Описание структур данных

**Класс ArrList:**

template< class T > - шаблон класса ArrList

T\* arMas; - указатель для выделения памяти под массив элементов типа T;

int\* arIndex – указатель для выделения памяти под массив индексов;

int arMem, count, c, size – целочисленные значения: индекс текущего элемента, максимальное количество элементов, индекс текущего свободного слота, количество занятых мест в списке;

**Описание методов:**

|  |  |
| --- | --- |
| Метод: | Описание: |
| ArrList<T>::ArrList(const int n) | Конструктор с параметром для класса ArrList. n – максимальное количество элементов в списке. |
| ArrList<T>::ArrList(const ArrList<T>& A) | Конструктор копирования класса ArrList. |
| ArrList<T>::~ArrList() | Деструктор этого класса. |
| bool ArrList<T>::IsFull() | Метод проверки списка на полноту. |
| bool ArrList<T>::IsEmpty() | Метод проверки списка на пустоту. |
| void ArrList<T>::PutStart(const T& a) | Метод, позволяющий положить элемент в начало списка. |
| void ArrList<T>::PutEnd(const T& a) | Метод, который позволяет положить элемент в конец списка. |
| T ArrList<T>::GetStart() | Метод, который позволяет забрать элемент с удалением из начала списка. |
| T ArrList<T>::GetEnd() | Метод, который позволяет забрать элемент с удалением из конца списка. |

## 4.3.Описание алгоритмов

**Подробное описание некоторых методов**

Установка элемента в начало списка:

* Проверка списка на полноту;
* Если список не полон, то:
  + В массив элементов записывается элемент в текущую свободную позицию;
  + В массив указателей записывается в текущую свободную позицию индекс текущего элемента (т.е. уже предыдущего);
  + Текущему элементу присваивается значение текущего свободного элемента;
  + Количество занятых элементов увеличивается на единицу;
  + Происходит поиск следующего первого свободного места.

Забрать элемент с удалением из конца списка:

* Происходит проверка списка на пустоту и в случае чего выбрасывается исключение;
* Если в списке всего один элемент, то вызывается метод забора элемента из начала списка;
* Если в списке больше чем один элемент, то:
  + Создаются два целочисленных элемента, один – текущий элемент, второй – следующий по списку за ним элемент;
  + По этим двум элемента находится последний элемент в списке;
  + Индекс последнего элемента освобождается, а сам элемент возвращается пользователю;

# 5.Заключение

В заключении можно сказать, что все поставленные цели и задачи были выполнены, а именно: создан класс «ArrList» с реализованными методами добавления и доступа к элементам списка, а также написаны к ним тесты, и они успешно пройдены.

# 6.Литература

* Учебные материалы к учебному курсу «Методы программирования» - Гергель В.П.