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: список на массивах»**

**Выполнил:** студент группы 381706-1

Голубева А. С.

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

аспирант каф. МОСТ ИИТММ

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

Нижний Новгород

2018.

Содержание

[1. Введение 3](#_Toc943229)

[2. Постановка задачи 4](#_Toc943230)

[3. Руководство пользователя 5](#_Toc943231)

[4. Руководство программиста 6](#_Toc943232)

[4.1 Описание структуры программы 6](#_Toc943233)

[4.2 Описание структур данных 6](#_Toc943234)

[4.3 Описание алгоритмов 7](#_Toc943235)

[5. Заключение 8](#_Toc943236)

[6. Литература 9](#_Toc943237)

# **Введение**

Списком называется упорядоченное множество, состоящее из переменного числа элементов, к которым применимы операции включения, исключения. Список, отражающий отношения соседства между элементами, называется линейным.

Длина списка равна числу элементов, содержащихся в списке, список нулевой длины называется пустым списком. Списки представляют собой способ организации структуры данных, при которой элементы некоторого типа образуют цепочку. Для связывания элементов в списке используют систему указателей.

Целью данной лабораторной работы является реализация односвязного списка на массивах.

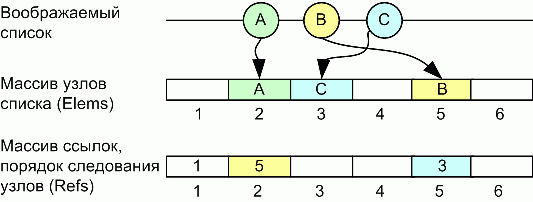


Рисунок 1. Представление списка на массивах

# **Постановка задачи**

Выполнение работы предполагает решение следующих задач:

1. Реализация класса TArrayList.
2. Реализация класса TException для обработки исключений
3. Реализация тестов и обеспечение их работоспособности.
4. Пример использования и обеспечение его работоспособности.

# **Руководство пользователя**

Рассмотрим пример работы программы.

Создается список, выполняются операции положить в начало 2, положить в начало 7 и положить 9 в конец списка. Затем выводится на экран результат работы операций забрать элемент из начала списка и забрать элемент из конца списка.

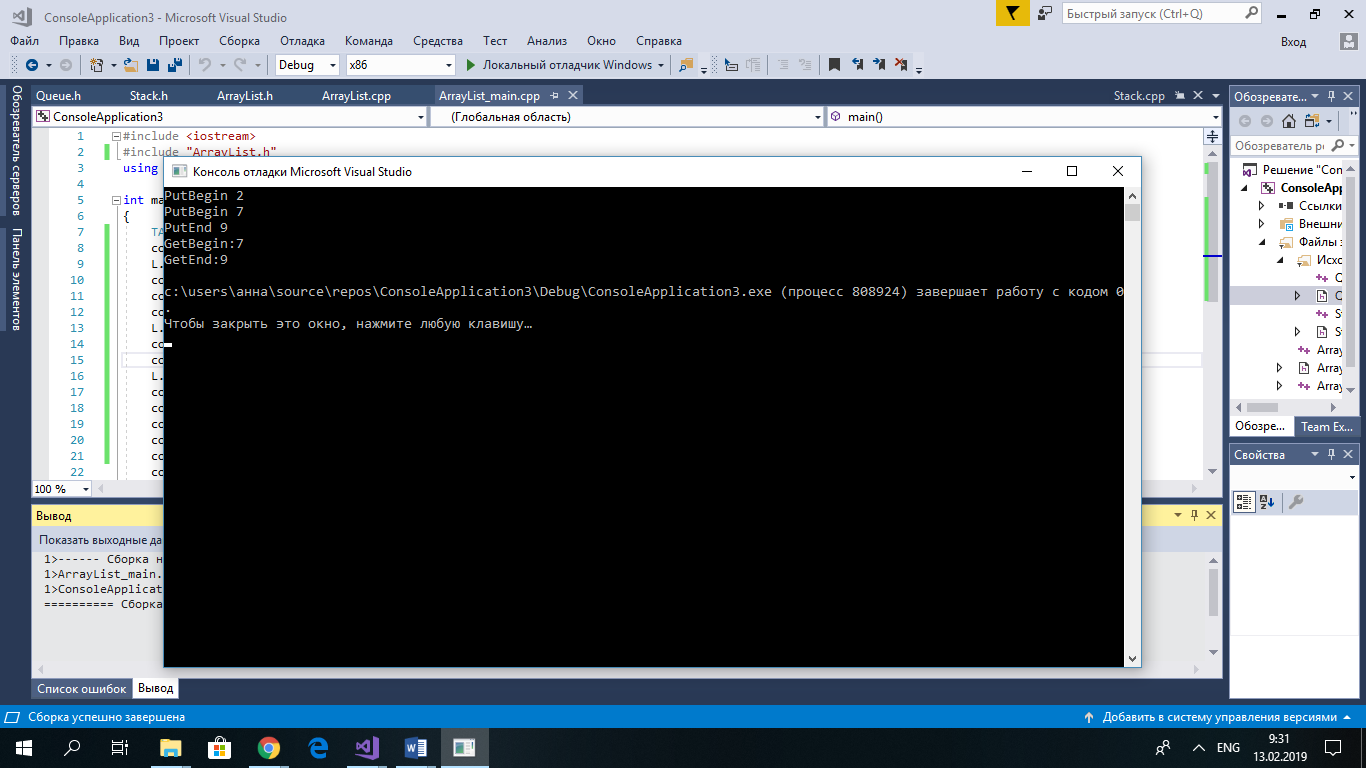


Рисунок 2. Пример работы программы

# **Руководство программиста**

## **Описание структуры программы**

Программа состоит из следующих модулей:

* Модуль ArrayListLib - содержит заголовочный файл ArrayList.h, в котором определен интерфейс класса TArrayList и реализованы его методы. (Класс TArrayList шаблонный), и файл TArrayList.cpp;
* Модуль ArrayListTest - содержит файл ArrayList\_Test.cpp, в котором реализованы тесты для класса TArrayList;
* Модуль ArrayList - содержит файл ArrayList\_main.cpp, в котором реализован пример использования класса TArrayList;
* Модуль Exception – содержит заголовочный файл Exception.h, в котором реализован класс исключений, и файл Exception.cpp

## **Описание структур данных**

**Класс TArrayList**

Класс TArrayList – шаблонный класс.

**Private:**

T \* mas - массив элементов списка.

int \*nextIndex - индексы, указывающие на следующий элемент списка.

int \*prevIndex - индексы, указывающие на предыдущий элемент списка.

int size - размер списка.

int count - количество элементов в списке.

int begin - индекс первого элемента списка.

int end - индекс последнего элемента списка.

TQueue <int> freeElem - Очередь свободных элементов.

**Public:**

TArrayList(int \_size = 10) - конструктор.

TArrayList(TArrayList<T> &A) - конструктор копирования.

void PutBegin(T elem) - метод, позволяющий положить элемент в начало списка.

void PutEnd(T elem) - метод, позволяющий положить элемент в конец списка.

T GetBegin() - метод, позволяющий забрать элемент из начала списка.

T GetEnd() - метод, позволяющий забрать элемент из конца списка.

bool IsFull() - проверка на полноту.

bool IsEmpty() - проверка на пустоту.

## **Описание алгоритмов**

**Добавление элемента в начало и в конец.**

Если список не полон, то создаем переменную free и присваиваем значение, взятое из очереди свободных элементов. Далее присваиваем элементу массива mas с индексом free значение, которое хотим положить в список. Устанавливаем, что для этого элемента следующим является элемент с индексом begin. Если список был не пуст, то устанавливаем для первого элемента предыдущим добавленный элемент predIndex[begin] = free. Если список был пуст, то определяем, что добавленный элемент является и последним элементом в списке, индекс begin переопределяем на добавленный элемент: begin = free. Увеличиваем на 1 число элементов в списке count.

Аналогично с добавлением в конец.

**Удаление элемента из начала и конца.**

Если список не пуст, то записываем в переменную elem значение начального элемента списка. Добавляем в очередь свободных элементов освободившуюся позицию begin. Уменьшаем на 1 число элементов в списке count. Устанавливаем nextIndex[begin] = predIndex[begin] = -2 – пустые ячейки. Если после извлечения первого элемента, список не пуст, то predIndex[newbegin] = -1. Затем индекс begin переопределяем на newbegin: begin = newbegin.

Аналогично с удалением из конца.

# **Заключение**

В результате лабораторной работы была реализована структура хранения данных: односвязный список на базе массивов.

Успешно выполнены тесты, проверяющие работоспособность методов класса TArrayList. Приведен пример, демонстрирующий работу со стеком.

# **Литература**

* 1. Гергель В.П. Методические материалы по курсу «Методы программирования 2», 2015: URL: <http://www.itmm.unn.ru/files/2018/11/Primer-1.7.-Razrabotka-obshhego-predstavleniya-linejnogo-spiska.pdf>
  2. Реализация связанных списков на базе массивов. Электронный ресурс.

URL: <https://rsdn.org/article/alg/list.xml>

(Дата обращения: 09.02.2019)