

**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 данных список на массивах»

Выполнил: студент группы 381706-1
Митягина Дарья Сергеевна

_____ Подпись

Научный руководитель:
ассистент каф. МОСТ ИИТММ
_____ Лебедев И.Г

Нижний Новгород
2018.

Оглавление

1. Введение	2
2. Постановка задачи	3
3. Руководство пользователя	4
4. Руководство программиста.....	5
4.1. Описание структуры программы.	5
4.2. Описание структур данных.	5
4.3. Описание алгоритмов.....	6
5. Заключение.....	7
6. Список литературы.....	8

1. Введение

В данной лабораторной работе будет реализована такая структура как список на массивах.

Связный список – структура, элементами которой служат записи с одним и тем же форматом, связанные друг с другом логически с помощью указателей, хранящихся в самих элементах.

В списке каждый элемент состоит из двух полей: содержательного и поля указателя. Содержательное поле хранит данные. Поле указателя хранит адрес следующего элемента списка. По указателю получаем доступ к следующему элементу, от него к очередному и т.д. Должен быть указатель начала списка.

Основные операции со списками:

- 1) включение элемента
- 2) удаление элемента
- 3) определить, является ли список пустым/полным

2. Постановка задачи

В данной лабораторной работе необходимо разработать библиотеку для хранения и работы со списком на массивах.

Для этого нам нужно:

- Описать и реализовать класс списка (TArrayList).
- Протестировать класс TArrayList с помощью GT.
- Реализовать класс TException для обработки исключений.

3. Руководство пользователя

При запуске программы пользователю представляется простой пример использования списков.

Создается объект класса TArrayList, список. Затем в него кладутся значения и выводятся на экран.

```
An example of using the List will be implemented here
90
80
70
60
50
40
30
20
10
0
```

Рис. 1 – пример использования списка на массивах

4. Руководство программиста

4.1. Описание структуры программы.

Программа состоит из модулей:

1. ArrayList – содержит в себе файл реализации примера использования класса TArrayList.
2. ArrayListTest – содержит в себе файл ArrayListTest.cpp, в котором находится набор тестов, для проверки работоспособности класса TArrayList.
3. ArrayListLib – содержит в себе файлы TArrayList.h, в котором описан и реализован класс TArrayList.
4. QueueLib – содержит файл Queue.h, в котором реализован класс TQueue.
5. StackLib – содержит файл Stack.h, в котором реализован класс TStack.

4.2. Описание структур данных.

Класс TArrayList

Этот класс является шаблонным.

Рассмотрим private-часть:

1. T * mas;
2. int *nextIndex - индексы, указывающие на следующий элемент списка
3. int *prevIndex - индексы, указывающие на предыдущий элемент списка
4. int size - размер списка
5. int begin - первый элемент списка(точнее, его индекс)
6. int end - последний элемент списка(точнее, его индекс)
7. int count - количество элементов в списке
8. TQueue <int> FE - Очередь свободных элементов

Рассмотрим public-часть:

1. TArrayList(int _size = 10) - конструктор
2. TArrayList(TArrayList<T> &A) - конструктор копирования
3. void PutBegin(T element) - положить в начало списка
4. void PutEnd(T element) - положить в конец списка
5. T GetBegin() - забрать из начала списка
6. T GetEnd() - забрать из конца списка

7. `bool IsFull()` - проверка на полноту

8. `bool IsEmpty()` - проверка на пустоту

9. `T GetSmthFromTheInside(int n)` - забрать с n-ой позиции

10. `void PutSmthInside(T element, int n)` – установить значение элемента, находящегося на n-ой позиции

4.3. Описание алгоритмов

В данной части не будут рассматриваться тривиальные методы, внимание уделим лишь некоторым.

1. Добавление в начало

Если список не полон, то присваиваем новой переменной `free` значение, взятое из очереди свободных элементов.

Затем присваиваем элементу, имеющему индекс `free`, пришедшее значение.

Счетчик числа элементов инкрементируем.

Аналогично с добавлением в конец.

2. Получение из начала

Если список не пуст, то запоминаем в новой переменной `element` значение начального элемента. Сдвигаем начало на один элемент, пополняем очередь свободных элементов освобожденной позицией. Декрементируем счетчик числа элементов, возвращаем `element`.

Аналогично с получением из конца.

3. Проверки на полноту и пустоту

Осуществляются с помощью использования переменной, хранящей число элементов списка.

5. Заключение

В процессе работы над данной лабораторной работой мне удалось реализовать такую структуру хранения данных, как список на массивах.

Кроме того, было достигнуто более глубокое понимание принципов работы с данной структурой, принципов тестирования программы с помощью GT.

6. Список литературы

1. Гергель В.П. Методические материалы по курсу «Методы программирования 2», 2015. 2. Статья, посвященная теме списков [<https://prog-cpp.ru/data-ols/>]
2. Статья, посвященная теме списков [<https://prog-cpp.ru/data-ols/>]