

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ  
ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение  
высшего образования**

**«Национальный исследовательский**

**Нижегородский государственный университет им. Н.И. Лобачевского»  
(ННГУ)**

**Институт информационных технологий, математики и механики**

**Кафедра математического обеспечения и суперкомпьютерных  
технологий**

## **ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ**

**«Реализация класса мультистек»**

**Выполнил:** студент группы  
381706-2

Крюков Дмитрий Алексеевич

\_\_\_\_\_ Подпись

**Научный руководитель:**

ассистент каф. МОСТ ИИТММ

Лебедев Илья Геннадьевич

\_\_\_\_\_ Подпись

Нижний Новгород

2018

# Содержание

Содержание.....	2
1. Введение.....	3
2. Постановка задачи.....	4
3. Руководство пользователя.....	5
4. Руководство программиста.....	6
4.1 Описание структуры программы.....	6
4.2 Описание структур данных.....	6
4.3 Описание алгоритмов.....	7
5. Заключение.....	8
6. Литература.....	9

# 1. Введение

Стек — это структура данных, которая работает по принципу **FILO** (first in — last out; первый пришел — последний ушел).

В стеке элемент, который вошел самый первый — выйдет самым последним. Получается, если вы добавили три элемента в стек первым будет удален последний добавленный элемент.

Применение стека упрощает и ускоряет работу программы, так как идет обращение к нескольким данным по одному адресу.

Мультистек - Структура хранения нескольких стеков в общей памяти

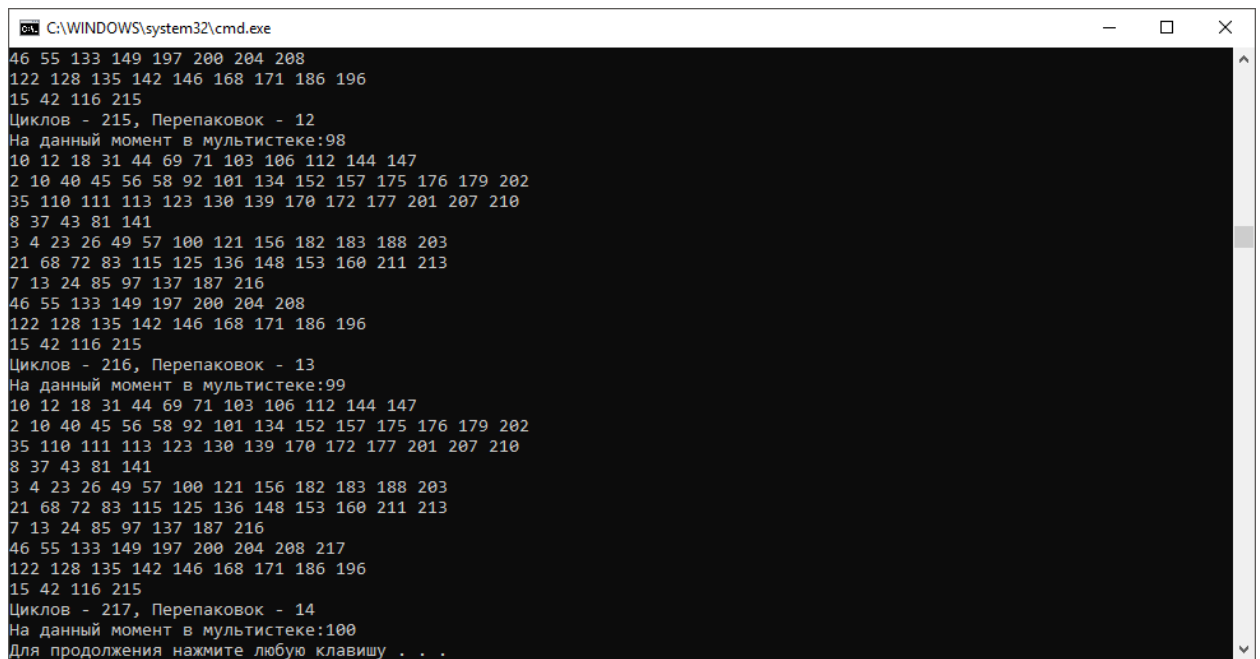
## 2. Постановка задачи

Реализация класса мультистек:

- Структура памяти для размещения нескольких стеков
- Начальное распределение памяти
- Ситуация локального переполнения памяти
- Оценка свободной памяти
- Динамическое перераспределение памяти при помощи перепакровки данных
- Схема наследования и последовательность разработки программ
- Реализация перепакровки

### 3. Руководство пользователя

Данная программа предназначена для тестирования динамической структуры мультистек. В мультистек поочередно загружаются или удаляются элементы до тех пор, пока их число не достигнет 100. После каждого действия на экран выводится мультистек, число элементов в нем и число перепакетов.



```
C:\WINDOWS\system32\cmd.exe
46 55 133 149 197 200 204 208
122 128 135 142 146 168 171 186 196
15 42 116 215
Циклов - 215, Перепакетов - 12
На данный момент в мультистеке:98
10 12 18 31 44 69 71 103 106 112 144 147
2 10 40 45 56 58 92 101 134 152 157 175 176 179 202
35 110 111 113 123 130 139 170 172 177 201 207 210
8 37 43 81 141
3 4 23 26 49 57 100 121 156 182 183 188 203
21 68 72 83 115 125 136 148 153 160 211 213
7 13 24 85 97 137 187 216
46 55 133 149 197 200 204 208
122 128 135 142 146 168 171 186 196
15 42 116 215
Циклов - 216, Перепакетов - 13
На данный момент в мультистеке:99
10 12 18 31 44 69 71 103 106 112 144 147
2 10 40 45 56 58 92 101 134 152 157 175 176 179 202
35 110 111 113 123 130 139 170 172 177 201 207 210
8 37 43 81 141
3 4 23 26 49 57 100 121 156 182 183 188 203
21 68 72 83 115 125 136 148 153 160 211 213
7 13 24 85 97 137 187 216
46 55 133 149 197 200 204 208 217
122 128 135 142 146 168 171 186 196
15 42 116 215
Циклов - 217, Перепакетов - 14
На данный момент в мультистеке:100
Для продолжения нажмите любую клавишу . . .
```

## 4. Руководство программиста

### 4.1 Описание структуры программы

1. Модуль multistacklib (TMultiStack.h, TMultiStack.cpp, Mstack.h) – реализация класса мультистек
2. Модуль multistack(main.cpp) – реализация программы для тестирования динамической структуры мультистек
3. Модуль multistacktest(multistack\_test.cpp) - тестирование класса мультистек при помощи Google C++ Testing Framework.

### 4.2 Описание структур данных

#### Структура

Класс MStack является наследником класса TStack и имеет дополнительные методы:

Методы:

MStack(ValType\*ind=NULL, int Size=0) — перегруженный конструктор

~Mstack() - перегруженный деструктор

void SetMem(ValType\* ind, int Size) — функция для управления памятью

класс TMultiStack :

#### Структура

Поля:

pStack - стеки - память выделяется из StackMem

MemSize - всего выделено памяти

CurrentCount - число элементов в мультистеке в данный момент

StackCount - число стеков

StackMem - память под стек

pStackMem - базовые адреса для памяти стеков

Методы:

`int IsEmpty(int ns) const` - контроль пустоты СД

`int IsFull(int ns) const` - контроль переполнения СД

`int operator==(const TMultiStack &ms)` - сравнение

`TMultiStack & operator= (const TMultiStack &ms)` - присваивание

`void Put(int ns, const ValType &Val)` - положить в стек

`ValType Get(int ns)` - взять из стека с удалением

## 4.3 Описание алгоритмов

**Добавление в стек n элемента a**

Если нет свободной памяти

Ошибка

Если стек n полон

Перепаковка

Положить в стек n значение a

## **5. Заключение**

В ходе работы была реализована структура памяти для размещения нескольких стеков- мультистек, в нем реализованы функции контроля пустоты и переполнения, добавление и извлечение элементов, а так же динамическое перераспределение памяти при помощи перепаковки данных



## 6. Литература

1. Гергель В.П. Методические материалы по курсу «Методы программирования 2», Нижний Новгород, 2015.
2. Стек | программирование на С и С++ - Режим доступа:  
<https://codelessons.ru/cplusplus/realizaciya-steka-stack-v-c.html>