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**

**«Структура данных - мультистек»**

**Выполнил:** студент группы 381706-1

Голованова Елена Александровна

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

ассистент каф. МОСТ ИИТММ

Лебедев Илья Геннадьевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись

Нижний Новгород

2018.

Оглавление

[1. Введение 3](#_Toc1351108)

[2. Постановка задачи 4](#_Toc1351109)

[3. Руководство пользователя 5](#_Toc1351110)

[4. Руководство программиста 6](#_Toc1351111)

[a. Описание структуры программы 6](#_Toc1351112)

[b. Описание структур данных 7](#_Toc1351113)

[c. Описание алгоритмов 9](#_Toc1351114)

[5. Заключение 10](#_Toc1351115)

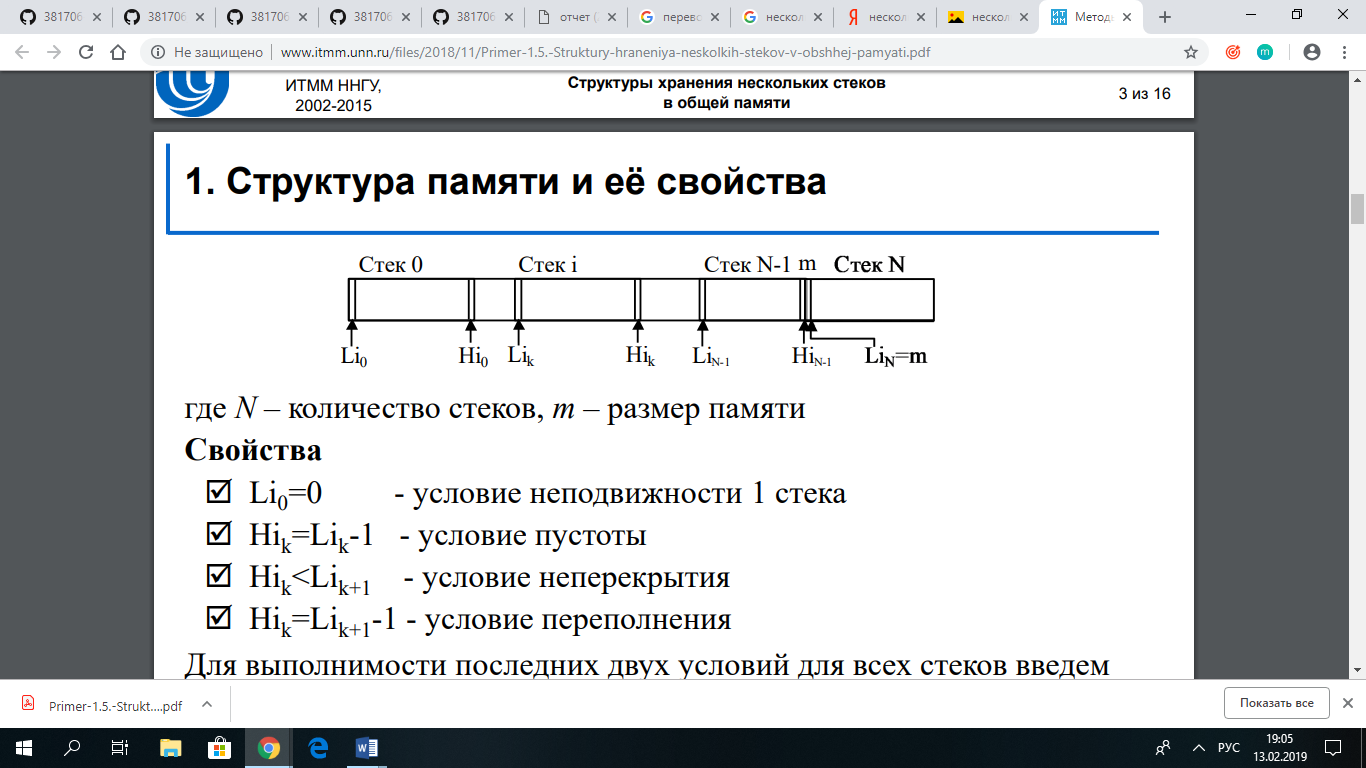
[6. Список литературы 11](#_Toc1351116)

# Введение

Перед программистом всегда стоит вопрос – как компактно использовать ресурсы компьютера. Одним из таких важных исчерпаемых ресурсов является память. Для ее экономии были придуманы разные структуры хранения данных.

Целью данной лабораторной работы является реализация структуры хранения мультистека.

Мультистек – структура хранения данных, которая представляет собой упорядоченный набор N стеков, фиксированного размера.



*Рис.1. Структура хранения мультистека*

На рисунке 1 отмечены :

* N - количество стеков,
* m - размер памяти,
* - индекс начала j-го стека,
* - индекс последнего элемента j-го стека.

Целью данной лабораторной работы является реализация структуры хранения данных мультистека.

# Постановка задачи

Выполнение работы предполагает решение следующих задач:

1. Реализация вспомогательного класса TNewStack, а так же класса мультистека TMltStack.
2. Пример программы, демонстрирующая работу класса TMltStack.
3. Реализация простых тестов на базе Google Test.

# 3. Руководство пользователя

Работа программы выглядит следующим образом:

1. Создается мультистек с определенными данными: размер мультистека - 12 элементов, количество стеков в мультистеке - 2.
2. Затем мультистек заполняется числами от 1 до 10: числа от 1 до 5 помещаются в первый стек, 6 – 10 во второй.
3. После этого на экране выводится мультистек.
4. Далее происходит извлечение элементов их стека.
5. Затем осуществляется попытка перепаковки мультистека путем добавления элемента «10» в 1-й стек.

# 4. Руководство программиста

## Описание структуры программы

Для реализации лабораторной работы создается несколько модулей:

* Модуль StackLib

Этот модуль включает в себя следующие файлы: Stack.h – в этом файле определен интерфейс класса стека и его реализация– и Stack.cpp – подключение заголовочного файла Stack.h

* Модуль Stack

Этот модуль содержит пример использования стека.

* Модуль StackTest

Этот модуль содержит тесты, проверяющие работу каждого методов класса TStack.

* Модуль Exception

Этот модуль содержит класс исключений.

* Модуль MultiStackLib

Этот модуль включает в себя следующие файлы: MultiStack.h – в этом файле определен интерфейс класса мультистека и его реализация– и MultiStack.cpp – подключение заголовочного файла MultiStack. Также в нем есть файл TNewStack.h, в котором описан интерфейс и реализация класса TNewStack.

* Модуль MultiStack

Этот модуль содержит пример использования мультистека.

* Модуль MultiStackTest

Этот модуль содержит тесты, проверяющие работу каждого методов класса MultiStack.

## Описание структур данных

Класс TStack является шаблонным.

* + 1. В зоне protected определены три поля и два метода:
* int Size – размер стека
* int Top – элемент, расположенный на вершине стека
* T\* Mas – элементы стека

2. В зоне public:

* int GetSize() { return Size; } – Возврат размера стека.
* TStack(int n = 0) – Конструктор инициализации.
* TStack(TStack &S) – Конструктор копирования.
* T Get() – Возврат элемента, расположенного на вершине стека.
* void PrintStack() – Вывод стека на экран.
* void Put(T A) – Добавление элемента на вершину стека, если стек не полон.
* bool IsFull() – Проверка на полноту.
* bool IsEmpty() – Проверка на пустоту. Стек пуст, если в нем нет ни одного элемента, т.е. когда количество элементов равно нулю.
* int operator!=(const TStack& stack) const; – Принимает ссылку на объект класса TStack, выполняет проверку на неравенство.
* int operator==(const TStack& stack) const; – Принимает ссылку на объект класса TStack, выполняет проверку на равенство.
* TStack& operator=(const TStack& stack); – Принимает ссылку на объект класса TStack, приравнивает исходный объект к полученному.

Класс TNewStack является шаблонным. Он наследуется от класса TStack

1. В зоне public:

* TNewStack(int \_size, T\* \_mas) – Конструктор по умолчанию.
* TNewStack(TNewStack<T>& A*)* – Конструктор копирования.
* int GetFreeMem() – Получение количества свободных ячеек в стеке.
* T Get() – Возврат элемента.
* void Push(T \_A*)* – Установить элемент в стек.
* int GetSize() – Получение размера стека.
* int GetTop( ) – Получение позиции вершины стека.
* void SetMas(int \_size, T\* \_mas) – Преобразование массива mas размером size в стек.
* void PrintNewStack() – Вывод стека.

Класс TMltStack является шаблонным.

1. В зоне protected:

* int size – Размер мультистека.
* T\* mas – Указатель на область память под хранение мультистека.
* int n – Количество стеков в мультистеке.
* TNewStack<T>\*\* stackMas – Массив указателей на начало каждого стека в мультистеке.
* int NumFree() – Количество свободных элементов в мультистеке.
* void Repack(int \_n) – Перепаковка стека с увеличением свободной памяти в n-м стеке.
  + 1. В зоне public:
* TMStack(int \_n = 1, int \_size = 10) – Конструктор по умолчанию.
* TMStack(TMStack &A) – Конструктор копирования.
* int GetSize() – Возврат размера мультистека.
* void Set(int \_n, T \_elem) – Установление в n-й стек элемент elem.
* T Get(int \_n) – Взятие элемента из n-го стека.
* bool IsFull(int \_n) – Проверка на полноту n-го стека.
* bool IsEmpty(int \_n) – Проверка на пустоту n-го стека.
* void PrintMStack() – Вывод мультистека.

## Описание алгоритмов

Здесь будет рассмотрен алгоритм перепаковки мультистека.

К нему необходимо прибегать, когда нам нужно положить новый элемент в стек, который уже полон, но в других есть хотя бы одна свободная ячейка.

Опишем алгоритм перепаковки мультистека относительно k-го стека:

* + - 1. Количество свободных ячеек, которые можно добавить в каждый стек add\_ev находим путем деления количества свободных ячеек во всем мультистеке free на количество стеков n:

|  |  |
| --- | --- |
|  |  |

* + - 1. Увеличиваем старые размеры size[i] стеков на add\_ev, получаем новые размеры стеков new\_size[i]:

|  |  |
| --- | --- |
|  |  |

* + - 1. Если количество свободных ячеек во всем мультистеке free не кратно количеству стеков n, то оставшиеся свободные ячейки добавляем в k-й стек:

|  |  |
| --- | --- |
|  |  |

* + - 1. Определяем новый индекс начала каждого стека new\_ind[i], с учетом их размера new\_size[i]:

|  |  |
| --- | --- |
|  |  |
|  |  |

* + - 1. Перемещение элементов стеков на новые позиции:
  1. Если индекс нового начала i-го стека new\_ind[i] не больше, чем индекс старого начала i-го стека ind[i], то копируем элементы по порядку, в котором они хранятся в старом стеке.
  2. Иначе идем по новым позициям стеков до тех пор, пока не выполняется 5.1. Затем копируем элементы, в котором они хранятся в старом стеке, но в обратном порядке. Иначе произойдет затирание памяти.

# 5. Заключение

В результате выполнения лабораторной работы была детально изучена и разработана структура хранения данных – мультистек. Эта структура была разработана с помощью наследования класса стека, вспомогательного класса и исключений.

Во время выполнения работы были получены новые знания и навыки.

Были реализованы тесты для проверки работоспособности вышеперечисленных классов на базе GoogleTest.

# 6. Список литературы

1. Википедия. Статья «Стек»

[https://neerc.ifmo.ru/wiki/index.php?title=%D0%A1%D1%82%D0%B5%D0%BA]

1. Васильев А.Н. Самоучитель С++ с примерами и задачами. -СПб.: Наука и Техника, 2016. -480с.
2. Гергель В.П. Методические материалы по курсу «Методы программирования 2», Нижний Новгород, 2015.