МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего профессионального образования

**«Дальневосточный федеральный университет»**

**ШКОЛА ЕСТЕСТВЕННЫХ НАУК**

**Кафедра информационных систем управления**

**ЛАБОРАТОРНАЯ РАБОТА № 8 ЗАДАЧА № 5**

По дисциплине «Объектно-ориентированное программирование»

На тему «Динамические структуры в языке C++»

|  |
| --- |
| Выполнил студент гр. Б8219 |
| О.В. Константинов |
|  |
| Проверил старший преподаватель |
| Г.Л. Берёзкина |
|  |
| (зачтено/не зачтено) |

г. Владивосток

2016

# Аннотация

Данный отчет подготовлен в рамках задания по дисциплине «Объектно-ориентированное программирование». Отчет призван систематизировать знания об использовании и реализации классов в языке C++. Отчет содержит список выполненных заданий с реализацией на языке C++ в среде программирования MS Visual Studio 2016.

# Задача 5

## Файл *“list\_stack\_lib.h”*

#pragma once

#include <cstddef>

template<typename T>

class List {

private:

struct Node;

public:

class Iterator {

public:

Iterator(Node\* node) : m\_node(node) { }

bool operator == (const Iterator& other) const {

if (this == &other) return true;

return m\_node == other.m\_node;

}

bool operator != (const Iterator& other) const {

return !operator == (other);

}

T operator\*() const {

if (m\_node) return m\_node->m\_t;

return T();

}

void operator++() {

if (m\_node) m\_node = m\_node->m\_next;

}

private:

Node\* m\_node;

};

public:

List() : m\_head(NULL) { };

~List() { while (m\_head) remove(); };

void append(const T& t);

void remove();

T head() const { return m\_head->m\_t; };

Iterator begin() const { return Iterator(m\_head); };

Iterator end() const { return Iterator(NULL); };

size\_t size() const;

private:

struct Node {

Node() : m\_next(NULL) { }

Node(const T& t) : m\_t(t), m\_next(NULL) { }

T m\_t;

Node\* m\_next;

};

Node\* m\_head;

};

template<typename T>

void List<T>::append(const T &t) {

if (Node\* node = new Node(t)) {

node->m\_next = m\_head;

m\_head = node;

}

}

template<typename T>

void List<T>::remove() {

if (m\_head) {

Node\* newHead = m\_head->m\_next;

delete m\_head;

m\_head = newHead;

}

}

template<typename T>

size\_t List<T>::size() const {

size\_t s = 0;

for (Iterator it = begin(); it != end(); ++it) ++s;

return s;

}

## Файл *“main.cpp”*

/\*

Задача 5 Создать шаблон класса «однонаправленный линейный список». Использовать его при решении задачи 3 из раздела «Динамические структуры данных» (см. задачи задания 4).

Задача 3. Используя очередь или стек, решить задачу: в файле записан текст сбалансированный по круглым скобкам. Требуется для каждой пары соответствующих открывающей и закрывающей скобок напечатать номера их позиций в тексте, упорядочив пары номеров по возрастанию номеров позиций:

а) закрывающих скобок (например, для текста а+(45-f(x)\*(b-c)) надо напечатать 8 10; 12 16; 3 17);

б) открывающих скобок ( например, для текста а+(45-f(x)\*(b-c)) надо напечатать 3 17; 8 10; 12 16).

\*/

#include "list\_stack\_lib.h"

#include <iostream>

#include <cstring>

using namespace std;

int main() {

List<int> lst;

List<pair<int, int>> lst3;

const string str = "а+(45-f(x)\*(b-c))";

for (size\_t i = 0; i < str.length(); i++)

if (str[i] == '(')

lst.append(i + 1);

else

if (str[i] == ')') {

cout << lst.head() << " " << i + 1 << "; ";

lst3.append(make\_pair(lst.head(), i + 1));

lst.remove();

}

cout << endl;

List<int> lst1, lst2;

for (size\_t i = 0; i < str.length(); i++)

if (str[i] == '(')

lst1.append(i + 1);

for (List<int>::Iterator it = lst1.begin(); it != lst1.end(); ++it)

lst2.append(\*it);

for (List<int>::Iterator it = lst2.begin(); it != lst2.end(); ++it)

for (List<pair<int, int>>::Iterator it1 = lst3.begin(); it1 != lst3.end(); ++it1)

if (\*it == (\*it1).first)

cout << (\*it1).first << " " << (\*it1).second << "; ";

cout << endl;

return 0;

}