|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

**Отчет по лабораторной работе**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Структуры и алгоритмы обработки данных** | | | | |
| *( наименование дисциплины )* | | | | |
| Направление подготовки | | | 09.03.01 Информатика и вычислительная техника | |
|  | | | |  |
| Тема | Линейный однонаправленный список (Лабораторная работа №1) | | | |
| Институт | | Информационных технологий (ИТ) | | |

Выполнил студент 3 курса

Группа ИВБО-06-17

Фамилия Имя Галеев Дмитрий

Проверил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Оглавление

[Оглавление 2](#_Toc27519326)

[1. Задание 1 5](#_Toc27519327)

[1.1 Вариант задания 5](#_Toc27519328)

[1.2 Абстрактный тип данных (далее АТД) для варианта задания, включая список общих функций из задания. 5](#_Toc27519329)

[1.3 Реализация АТД 5](#_Toc27519330)

[1.4 Таблица тестов 7](#_Toc27519331)

[1.5 Скриншоты выполнения тестов 8](#_Toc27519332)

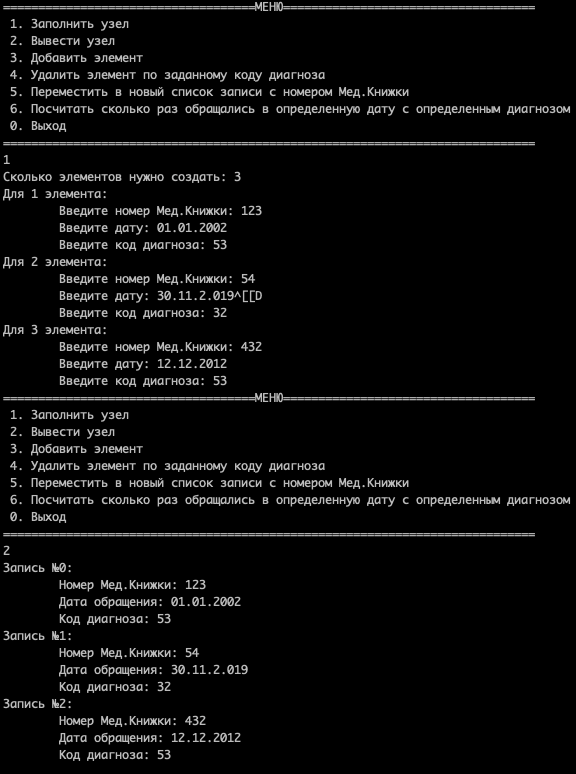
[1.6 Код основной программы с комментариями 11](#_Toc27519333)

[2. Задание 2 17](#_Toc27519334)

[2.1 Структура узла 17](#_Toc27519335)

[2.2 Структура класса 17](#_Toc27519336)

[2.3 Таблица тестов 17](#_Toc27519337)

[2.4 Скриншоты выполнения тестов  19](#_Toc27519338)

[2.5 Код основной программы с комментариями 21](#_Toc27519339)

[Заключение 28](#_Toc27519340)

# Задание 1

## Вариант задания

Разработать многомодульную программу, которая демонстрирует выполнение всех операций, определённых вариантом, над линейным однонаправленным динамическим списком.

Разработайте функции для выполнения операции над линейным списком:

* создание списка из n узлов
* вывод списка
* поиск узла с заданным значением (операция должна возвращать указатель на заданный узел).

Оформите операции предложенных вариантов в виде функций и включите в отдельный файл с расширением cpp. Подключите к этому файлу заголовочный файл с определением структуры узла.

Разработайте программу, управляемую текстовым меню, согласно заданию и включите в меню демонстрацию выполнения всех операций задания и варианта.

## Абстрактный тип данных (далее АТД) для варианта задания, включая список общих функций из задания.

Номер мед. полиса, Дата обращения, Код диагноза (число).

Вставка нового узла перед первым узлом с заданным значением Мед. полиса, если такого нет, то узел вставить в конец списка. Удаление из списка всех узлов с заданным значением Кода диагноза. Переместить все узлы с одинаковым мед. полисом в новый список. Определить количество обращений в одну и туже дату с одним и тем же диагнозом.

## Реализация АТД

* Определение типа информационной части узла (typedef)

**Описать поля и методы без типов результатов и типов параметров**

Структура Пациент {

Номер медицинского полиса;

Дата обращения;

Код диагноза;

Указатель на следующий элемент;

Метод вставка нового элемента;

Метод удаления записи по номеру медицинского полиса;

Метод перемещения в новый список с одинаковым полисом;

Метод подсчитывания количества обращений;

}

* Структура узла

Номер медицинского полиса;

Дата обращения;

Код диагноза;

Указатель на следующий элемент;

* Алгоритмы задач варианта, которые не включены в АТД на псевдокоде и использующие операции из АТД.

Вывод всего списка:

Пока (есть головной элемент) {

Вывод номера записи;

Вывод номера мед полиса;

Вывод даты обращения;

Вывод кода диагноза;

}

Инициализация узла списка:

Создание указателя на переменную типа узла;

Запись пустого значения в поле дата;

Создание указателя на следующий узел со значением NULL;

Запись пустого значения в поле медицинский полис;

Запись пустого значения в поле код диагноза;

Добавление нового узла:

Создание указателя на переменную типа узла;

Запись значения в поле медицинский полис;

Запись значения в поле дата;

Запись значения в поле код диагноза;

Создание указателя на следующий узел со значением NULL;

Создание списка из n элементов:

Ввод количества элементов;

Пока (временная переменная не достигла значения n) {

Вызов метода инициализации узла;

Ввод номера медицинского полиса;

Ввод даты обращения;

Ввод кода диагноза;

Запись введенного значения в поле медицинский полис;

Запись введенного значения в поле дата;

Запись введенного значения в поле код диагноза;

}

* Список модулей реализации АТД (или описать где расположена реализация АТД)

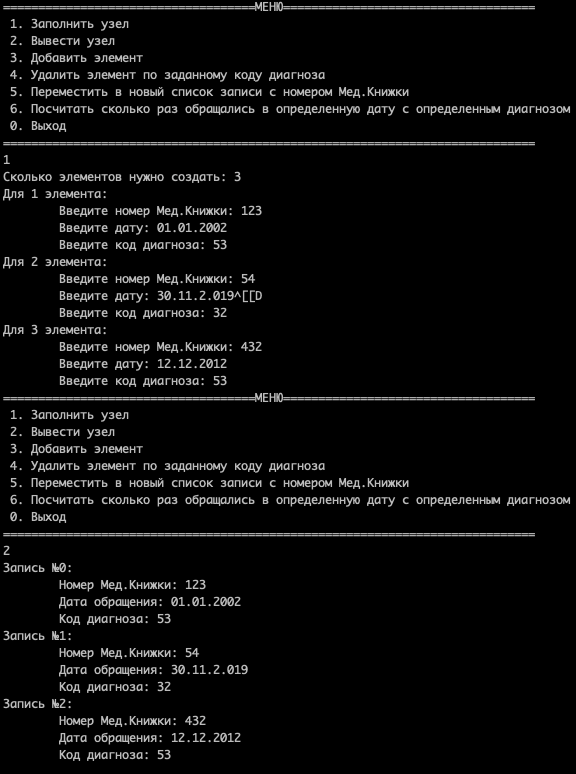
Для реализации использовались:

Структура s\_node

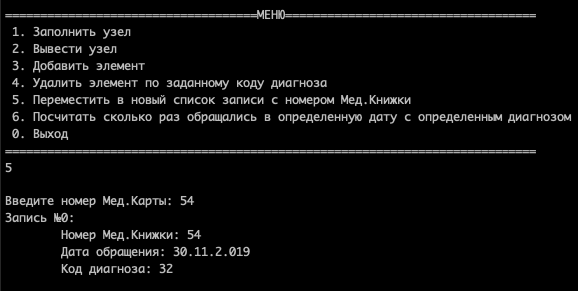
## Таблица тестов

|  |  |  |  |
| --- | --- | --- | --- |
| №  теста | Входные данные | Ожидаемый результат | Фактический результат |
| 1 | Сколько элементов нужно создать: 3  Для 1 элемента:  Введите номер Мед.Книжки: 123  Введите дату: 01.01.2002  Введите код диагноза: 53  Для 2 элемента:  Введите номер Мед.Книжки: 54  Введите дату: 30.11.2.019  Введите код диагноза: 32  Для 3 элемента:  Введите номер Мед.Книжки: 432  Введите дату: 12.12.2012  Введите код диагноза: 53 | Запись №0:  Номер Мед.Книжки: 123  Дата обращения: 01.01.2002  Код диагноза: 53  Запись №1:  Номер Мед.Книжки: 54  Дата обращения: 30.11.2.019  Код диагноза: 32  Запись №2:  Номер Мед.Книжки: 432  Дата обращения: 12.12.2012  Код диагноза: 53 | Совпадает с ожидаемым |
| 2 | Введите номер Мед.Карты: 54  Введите дату обращения: 12.02.2019  Введите код диагноза: 43 | Запись №0:  Номер Мед.Книжки: 123  Дата обращения: 01.01.2002  Код диагноза: 53  Запись №1:  Номер Мед.Книжки: 54  Дата обращения: 12.02.2019  Код диагноза: 43  Запись №2:  Номер Мед.Книжки: 54  Дата обращения: 30.11.2.019  Код диагноза: 32  Запись №3:  Номер Мед.Книжки: 432  Дата обращения: 12.12.2012  Код диагноза: 53 | Совпадает с ожидаемым |
| 3 | Введите код диагноза: 43 | Запись №0:  Номер Мед.Книжки: 123  Дата обращения: 01.01.2002  Код диагноза: 53  Запись №1:  Номер Мед.Книжки: 54  Дата обращения: 30.11.2.019  Код диагноза: 32  Запись №2:  Номер Мед.Книжки: 432  Дата обращения: 12.12.2012  Код диагноза: 53 | Совпадает с ожидаемым |
| 5 | Введите номер Мед.Карты: 54 | Запись №0:  Номер Мед.Книжки: 54  Дата обращения: 30.11.2.019  Код диагноза: 32 | Совпадает с ожидаемым |

## Скриншоты выполнения тестов







## Код основной программы с комментариями

Файл Lab1z1.h

#pragma once

#include <stdlib.h>

#include <iostream>

#include <string>

typedef struct s\_node

{

unsigned id;

std::string date;

unsigned code;

struct s\_node \*next;

} t\_node;

t\_node \*push\_node(t\_node\* head);

void delete\_by\_code(t\_node\*& head);

t\_node\* move\_to\_new(t\_node\* head);

void count\_codes(t\_node\* head);

Файл Lab1z1.cpp

#include "Lab1.h"

#include <iostream>

#include <string>

using namespace std;

// вывод списка

void print\_node(t\_node\* head)

{

t\_node\* tmp;

unsigned i;

i = 0;

tmp = head;

if (!tmp)

std::cout << "Вы не заполнили список" << endl;

while (tmp)

{

cout << "Запись №" << i << ": " << endl;

cout << "\tНомер Мед.Книжки: " << tmp->id << endl;

cout << "\tДата обращения: " << tmp->date << endl;

cout << "\tКод диагноза: " << tmp->code << endl;

if (!tmp->next)

cout << "\n";

tmp = tmp->next;

i++;

}

}

t\_node\* node\_init(void)

{

t\_node\* res;

if (!(res = new t\_node))

exit(-1);

res->date = "";

res->next = NULL;

res->id = 0;

res->code = 0;

return (res);

}

t\_node\* add\_new(unsigned id, string date, unsigned code)

{

t\_node\* res;

if (!(res = new t\_node))

exit(-1);

res->id = id;

res->date = date;

res->code = code;

res->next = NULL;

return (res);

}

// Создание списка из n узлов

t\_node\* creatennode(void)

{

t\_node\* head;

t\_node\* tmp;

size\_t i;

size\_t n;

unsigned id;

std::string date;

unsigned code;

std::cout << "Сколько элементов нужно создать: ";

std::cin >> n;

i = 0;

head = NULL;

tmp = NULL;

while (++i <= n)

{

if (!head)

{

head = node\_init();

std::cout << "Для " << i << " элемента:\n";

std::cout << "\tВведите номер Мед.Книжки: ";

std::cin >> id;

std::cout << "\tВведите дату: ";

std::cin >> date;

std::cout << "\tВведите код диагноза: ";

std::cin >> code;

head->id = id;

head->date = date;

head->code = code;

tmp = head;

}

else

{

std::cout << "Для " << i << " элемента:\n";

std::cout << "\tВведите номер Мед.Книжки: ";

std::cin >> id;

std::cout << "\tВведите дату: ";

std::cin >> date;

std::cout << "\tВведите код диагноза: ";

std::cin >> code;

tmp->next = add\_new(id, date, code);

tmp = tmp->next;

}

}

return (head);

}

int main(void)

{

t\_node\* head;

string str;

int c;

int f\_m;

c = 1;

f\_m = 1;

head = NULL;

while (c) {

if (f\_m) //Для того, чтобы меню выводилось 1 раз мы вводим условие

{

std::cout << "====================================МЕНЮ====================================" << endl;

std::cout << " 1. Заполнить узел" << endl

<< " 2. Вывести узел" << endl

<< " 3. Добавить элемент" << endl

<< " 4. Удалить элемент по заданному коду диагноза" << endl

<< " 5. Переместить в новый список записи с номером Мед.Книжки" << endl

<< " 6. Посчитать сколько раз обращались в определенную дату с определенным диагнозом" << endl

<< " 0. Выход" << endl;

std::cout << "============================================================================" << endl;

}

else

std::cout << "Введите пункт меню: ";

std::cin >> c;

switch (c) {

case 1:

head = creatennode();

break;

case 2:

print\_node(head);

break;

case 3:

head = push\_node(head);

break;

case 4:

delete\_by\_code(head);

break;

case 5:

print\_node(move\_to\_new(head));

break;

case 6:

count\_codes(head);

break;

case 0:

exit(1);

break;

}

}

return (0);

}

Файл Functions.cpp

#include "Lab1.h"

t\_node \*push\_node(t\_node\* head)

{

t\_node\* res;

t\_node\* node;

unsigned id;

std::cout << "Введите номер Мед.Карты: ";

std::cin >> id;

res = new t\_node;

res->id = id;

node = head;

while (node->next && node->next->id != id)

node = node->next;

res->next = node->next;

node->next = res;

std::cout << "\tВведите дату обращения: ";

std::cin >> res->date;

std::cout << "\tВведите код диагноза: ";

std::cin >> res->code;

return (head);

}

void delete\_by\_code(t\_node\*& head)

{

t\_node\* cur;

t\_node\* prev;

unsigned code;

std::cout << "\nВведите код диагноза: "; std::cin >> code;

cur = head;

prev = NULL;

while (cur)

{

if (cur->code == code)

{

if (prev)

{

prev->next = cur->next;

delete cur;

cur = prev->next;

}

else

{

head = head->next;

delete cur;

cur = head;

}

}

else

{

prev = cur;

cur = cur->next;

}

}

}

t\_node\* move\_to\_new(t\_node\* head)

{

t\_node\* node;

t\_node\* res;

t\_node\* prev;

unsigned code;

std::cout << "\nВведите номер Мед.Карты: ";

std::cin >> code;

node = head;

res = 0;

while (node && node->id != code)

{

prev = node;

node = node->next;

}

res = node;

while (node && node->id == code)

node = node->next;

prev->next = node;

node = res;

while (node && node->next != prev->next)

node = node->next;

node->next = 0;

return (res);

}

void count\_codes(t\_node\* head)

{

t\_node\* tmp;

unsigned counter;

unsigned code;

std::string data;

counter = 0;

std::cout << "\nВведите код диагноза: "; std::cin >> code;

std::cout << "Введите дату обращения: "; std::cin >> data;

tmp = head;

while (tmp)

{

if (tmp->code == code && tmp->date == data)

counter++;

tmp = tmp->next;

}

std::cout << "Количество обращений: " << counter << std::endl;

}

# Задание 2

## Структура узла

Узел Пациент:

Номер медицинского полиса;

Дата обращения;

Код диагноза;

Указатель на следующий элемент

## Структура класса

class Tnode

{

private:

unsigned id; // Номер мед карты

std::string date; // Дата обращения

unsigned code; // Код диагноза

Tnode\* next;

public:

Tnode(unsigned i, std::string d, unsigned c, Tnode\* n);

Tnode();

void move\_data(Tnode\* n);

unsigned get\_id();

std::string get\_date();

unsigned get\_code();

Tnode\* get\_next();

void set\_next(Tnode\* n);

void push\_node();

void delete\_by\_code();

Tnode\* move\_to\_new();

void count\_codes();

void print\_node();

void createnode();

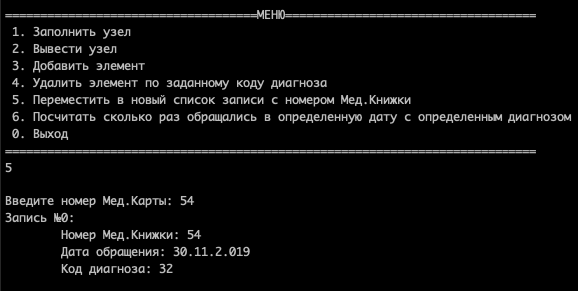
};

## Таблица тестов

|  |  |  |  |
| --- | --- | --- | --- |
| №  теста | Входные данные | Ожидаемый результат | Фактический результат |
| 1 | Сколько элементов нужно создать: 3  Для 1 элемента:  Введите номер Мед.Книжки: 123  Введите дату: 01.01.2002  Введите код диагноза: 53  Для 2 элемента:  Введите номер Мед.Книжки: 54  Введите дату: 30.11.2.019  Введите код диагноза: 32  Для 3 элемента:  Введите номер Мед.Книжки: 432  Введите дату: 12.12.2012  Введите код диагноза: 53 | Запись №0:  Номер Мед.Книжки: 123  Дата обращения: 01.01.2002  Код диагноза: 53  Запись №1:  Номер Мед.Книжки: 54  Дата обращения: 30.11.2.019  Код диагноза: 32  Запись №2:  Номер Мед.Книжки: 432  Дата обращения: 12.12.2012  Код диагноза: 53 | Совпадает с ожидаемым |
| 2 | Введите номер Мед.Карты: 54  Введите дату обращения: 12.02.2019  Введите код диагноза: 43 | Запись №0:  Номер Мед.Книжки: 123  Дата обращения: 01.01.2002  Код диагноза: 53  Запись №1:  Номер Мед.Книжки: 54  Дата обращения: 12.02.2019  Код диагноза: 43  Запись №2:  Номер Мед.Книжки: 54  Дата обращения: 30.11.2.019  Код диагноза: 32  Запись №3:  Номер Мед.Книжки: 432  Дата обращения: 12.12.2012  Код диагноза: 53 | Совпадает с ожидаемым |
| 3 | Введите код диагноза: 43 | Запись №0:  Номер Мед.Книжки: 123  Дата обращения: 01.01.2002  Код диагноза: 53  Запись №1:  Номер Мед.Книжки: 54  Дата обращения: 30.11.2.019  Код диагноза: 32  Запись №2:  Номер Мед.Книжки: 432  Дата обращения: 12.12.2012  Код диагноза: 53 | Совпадает с ожидаемым |
| 5 | Введите номер Мед.Карты: 54 | Запись №0:  Номер Мед.Книжки: 54  Дата обращения: 30.11.2.019  Код диагноза: 32 | Совпадает с ожидаемым |

## Скриншоты выполнения тестов





## Код основной программы с комментариями

Файл Tnode.h

#pragma once

#include <stdlib.h>

#include <iostream>

#include <string>

using namespace std;

class Tnode

{

private:

unsigned id;

std::string date;

unsigned code;

Tnode\* next;

public:

Tnode(unsigned i, std::string d, unsigned c, Tnode\* n);

Tnode();

void move\_data(Tnode\* n);

unsigned get\_id();

std::string get\_date();

unsigned get\_code();

Tnode\* get\_next();

void set\_next(Tnode\* n);

void set\_id(unsigned i);

void push\_node();

void delete\_by\_code();

Tnode\* move\_to\_new();

void count\_codes();

void print\_node();

void createnode();

};

Файл Tnode.cpp

#include "Tnode.h"

unsigned Tnode::get\_id()

{

return (this->id);

}

std::string Tnode::get\_date()

{

return (this->date);

}

unsigned Tnode::get\_code()

{

return (this->code);

}

Tnode\* Tnode::get\_next()

{

return (this->next);

}

void Tnode::set\_next(Tnode\* n)

{

this->next = n;

}

void Tnode::set\_id(unsigned i)

{

id = i;

}

Tnode::Tnode()

{

id = 0;

date = "";

code = 0;

next = 0;

}

Tnode\* add\_new(unsigned id, string date, unsigned code)

{

Tnode\* res;

if (!(res = (new Tnode(id, date, code, NULL))))

exit(-1);

return (res);

}

Tnode::Tnode(unsigned i, std::string d, unsigned c, Tnode\* n)

{

id = i;

this->date = d;

this->code = c;

this->next = n;

}

void Tnode::move\_data(Tnode\* n)

{

unsigned id;

std::string date;

unsigned code;

Tnode\* tmp;

id = this->id;

date = this->date;

code = this->code;

tmp = this->next;

this->id = n->get\_id();

this->date = n->get\_date();

this->code = n->get\_code();

delete n;

n = new Tnode(id, date, code, tmp);

this->next = n;

}

void Tnode::push\_node()

{

unsigned id;

Tnode \*res;

Tnode\* node;

std::cout << "Введите номер Мед.Карты: ";

std::cin >> id;

res = add\_new(id, "\0", 0);

node = this;

while (node->get\_next() && node->get\_next()->id != id)

node = node->get\_next();

res->set\_next(node->get\_next());

node->set\_next(res);

std::cout << "\tВведите дату обращения: ";

std::cin >> res->date;

std::cout << "\tВведите код диагноза: ";

std::cin >> res->code;

}

void Tnode::delete\_by\_code()

{

Tnode\* cur;

Tnode\* prev;

unsigned code;

std::cout << "\nВведите код диагноза: "; std::cin >> code;

cur = this;

prev = NULL;

while (cur)

{

if (cur->code == code)

{

if (prev)

{

prev->next = cur->next;

delete cur;

cur = prev->next;

}

else

{

move\_data(this->next);

delete cur;

cur = this;

}

}

else

{

prev = cur;

cur = cur->next;

}

}

}

Tnode\* Tnode::move\_to\_new()

{

Tnode\* node;

Tnode\* res;

Tnode\* prev;

unsigned code;

std::cout << "\nВведите номер Мед.Карты: ";

std::cin >> code;

node = this;

res = 0;

while (node && node->get\_id() != code)

{

prev = node;

node = node->get\_next();

}

res = node;

while (node && node->get\_id() == code)

node = node->get\_next();

prev->set\_next(node);

node = res;

while (node && node->get\_next() != prev->get\_next())

node = node->get\_next();

node->set\_next(0);

return (res);

}

void Tnode::count\_codes()

{

Tnode\* tmp;

unsigned counter;

unsigned code;

std::string data;

counter = 0;

std::cout << "\nВведите код диагноза: "; std::cin >> code;

std::cout << "Введите дату обращения: "; std::cin >> data;

tmp = this;

while (tmp)

{

if (tmp->code == code && tmp->date == data)

counter++;

tmp = tmp->next;

}

std::cout << "Количество обращений: " << counter << std::endl;

}

void Tnode::print\_node()

{

Tnode\* tmp;

unsigned i;

i = 0;

tmp = this;

if (!tmp)

std::cout << "Список пуст" << endl;

while (tmp)

{

cout << "Элемент №" << i << ": " << endl;

cout << "\tНомер Мед.Карты: " << tmp->id << endl;

cout << "\tДата обращщения: " << tmp->date << endl;

cout << "\tКод диагноза: " << tmp->code << endl;

if (!tmp->next)

cout << "\n";

tmp = tmp->get\_next();

i++;

}

}

Tnode\* node\_init(void)

{

Tnode\* res;

if (!(res = new Tnode(0, "\0", 0, NULL)))

exit(-1);

return (res);

}

void Tnode::createnode(void)

{

Tnode \*tmp;

size\_t i;

size\_t n;

unsigned id;

std::string date;

unsigned code;

std::cout << "Сколько элементов нужно создать: ";

std::cin >> n;

i = 0;

tmp = this;

while (++i <= n)

{

if (i == 1)

{

std::cout << "Для " << i << " элемента:\n";

std::cout << "\tВведите номер Мед.Книжки: ";

std::cin >> id;

std::cout << "\tВведите дату: ";

std::cin >> date;

std::cout << "\tВведите код диагноза: ";

std::cin >> code;

this->id = id;

this->date = date;

this->code = code;

tmp = this;

}

else

{

std::cout << "Для " << i << " элемента:\n";

std::cout << "\tВведите номер Мед.Книжки: ";

std::cin >> id;

std::cout << "\tВведите дату: ";

std::cin >> date;

std::cout << "\tВведите код диагноза: ";

std::cin >> code;

tmp->set\_next(add\_new(id, date, code));

tmp = tmp->get\_next();

}

}

}

Файл lab1z2.cpp

// lab1z2.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.

//

#include <iostream>

#include "Tnode.h"

int main(void)

{

Tnode node;

string str;

int c;

int f\_m;

c = 1;

f\_m = 1;

// node = new Tnode(0, 0, 0, 0);

while (c) {

if (f\_m) //Для того, чтобы меню выводилось 1 раз мы вводим условие

{

std::cout << "====================================МЕНЮ====================================" << endl;

std::cout << " 1. Заполнить узел" << endl

<< " 2. Вывести узел" << endl

<< " 3. Добавить элемент" << endl

<< " 4. Удалить элемент по заданному коду диагноза" << endl

<< " 5. Переместить в новый список записи с номером Мед.Книжки" << endl

<< " 6. Посчитать сколько раз обращались в определенную дату с определенным диагнозом" << endl

<< " 0. Выход" << endl;

std::cout << "============================================================================" << endl;

}

else

std::cout << "Введите пункт меню: ";

std::cin >> c;

switch (c) {

case 1:

node.createnode();

break;

case 2:

node.print\_node();

break;

case 3:

node.push\_node();

break;

case 4:

node.delete\_by\_code();

break;

case 5:

Tnode \* tmp;

tmp = node.move\_to\_new();

tmp->print\_node();

tmp = NULL;

break;

case 6:

node.count\_codes();

break;

case 0:

exit(1);

break;

}

}

return (0);

}

# Заключение

В данной лабораторной работе мы научились создавать односвязные элементарные списки посредством реализации на элементарных структурах и классах с использованием ООП на языке C++.

Для выполнения активно использовались указатели на объекты линейного списка с помощью которых, реализовывалась связь между элементами списка.

Полученные навыки и знания дают возможность выполнения последующих лабораторных работ и заданий, а так же создание собственных односвязных списков.