Автор: Татаренко А., КІТ-119а

Дата: 17 червня 2020

Лабораторна робота №13.

АЛГОРИТМИ ПЕРЕМІЩЕННЯ ТА ПОШУКУ

Тема. STL. Алгоритми переміщення та пошуку.

Мета – на практиці порівняти STL-алгоритми, що не модифікують послідовність.

1 Завдання до роботи

Індивідуальне завдання 19.

Поширити попередню лабораторну роботу, додаючи такі можливості діалогового меню:
- виведення всіх елементів масиву за допомогою STL-функції for_each;
- визначення кількості елементів за заданим критерієм;
- пошук елемента за заданим критерієм.

2 Розробка алгоритму розв’язання задачі.

2.1 Опис змінних

Arr stud_array; class Student; class Arr;

Класи, методи, функції, конструктори

3 Код програми

class1.h

```
#pragma once
#include "Header.h"

class Student
{
protected:
	int age;
	int number_stud;
	int middle_mark;
	string name;
	bool debt;
	int prog_d;

public:
	virtual string get_info() const;
	virtual stringstream get_str() const;
	int get_numb() const;
	virtual bool elementOutput(int, string);
	virtual int countElement(int, string);

	Student();
	Student(int, int, int, string, bool, int);
	Student(const Student&);
	virtual ~Student();

	friend ostream& operator<< (ostream&, const Student&);
	virtual bool operator==(const int) const;

};
```

class2.h

```
#pragma once
#include "class1.h"

class Course final : public Student
{
private:
	int course;

public:
	string get_info() const override final;
	stringstream get_str() const override final;
	bool elementOutput(int, string) override final;
	int countElement(int, string) override final;

	Course();
	Course(int, int, int, string, bool, int, int);
	Course(const Course&);
	~Course() override final;

	bool operator==(const int) const override final;

};
```

Header.h

```
#pragma once

#define _CRT_SECURE_NO_WARNINGS
```

```
#define CRTDBG_MAP_ALLOC
#include <crtdbg.h>
#define DEBUG_NEW new(_NORMAL_BLOCK, FILE, __LINE)

#define _CRT_SECURE_NO_WARNINGS
#define CRTDBG_MAP_ALLOC
#include <crtdbg.h>
#define DEBUG_NEW new(_NORMAL_BLOCK, FILE, __LINE)

#include <string>
#include <iostream>
#include <iomanip>
#include <locale>
#include <fstream>
#include <sstream>
#include <istream>
#include <vector>
#include <memory>
#include <list>
#include <map>
#include <set>
#include <algorithm>

using std::string;
using std::cin;
using std::cout;
using std::endl;
using std::setw;
using std::boolalpha;
using std::setiosflags;
using std::ios;
using std::ifstream;
using std::ostream;
using std::ofstream;
using std::stringstream;
using std::istream;
using std::vector;
using std::list;
using std::map;
using std::set;
using std::unique_ptr;
using std::advance;
using std::stoi;
using std::for_each;
```

class1.cpp

```
#include "class1.h"

string Student::get_info() const
{
	stringstream temp;

	temp.setf(std::ios::left);
	temp << setw(10) << age << setw(8) << number_stud << setw(16) << middle_mark <<
setw(9)
		<< name << setw(7) << debt << setw(14) << prog_d;

	return temp.str();
}

int Student::get_numb() const
{
	return number_stud;
```

```
}

stringstream Student::get_str() const
{
	stringstream temp;
	temp << " " << age << " " << number_stud << " " << middle_mark << " "
		<< name << " " << debt << " " << prog_d;

	return temp;
}

int Student::countElement(int value, string data)
{
	try
	{
		if (value == 1)
		{
			if (this->name == data)
				return 1;
			else
				return 0;
		}
		else if (value == 2)
		{
			int number = stoi(data);
			if (this->age == number)
				return 1;
			else
				return 0;
		}
		else if (value == 3)
		{
			int number = stoi(data);
			if (this->middle_mark == number)
				return 1;
			else
				return 0;
		}
		else if (value == 4)
		{
			int number = stoi(data);
			if (this->prog_d == number)
				return 1;
			else
				return 0;
		}
		else if (value == 5)
		{
			int number = stoi(data);
			if (this->number_stud == number)
				return 1;
			else
				return 0;
		}
		else if (value == 6)
		{
			int number = 0;
			if (data == "true" || data == "true" || data == "1")
				number = 1;
			else
				number = 0;

			if (this->debt == number)
				return 1;
			else
```

```
                        return 0;
                }
        }
        catch (const std::exception& ex)
        {
                cout << ex.what() << endl;
                return 0;
        }

        return 0;
}

bool Student::elementOutput(int value, string data)
{
        try
        {
                if (value == 1)
                {
                        if (this->name == data)
                                cout << *this << endl;
                        return true;
                }
                else if (value == 2)
                {
                        int number = stoi(data);
                        if (this->age == number)
                                cout << *this << endl;
                        return true;
                }
                else if (value == 3)
                {
                        int number = stoi(data);
                        if (this->middle_mark == number)
                                cout << *this << endl;
                        return true;
                }
                else if (value == 4)
                {
                        int number = stoi(data);
                        if (this->prog_d == number)
                                cout << *this << endl;
                        return true;
                }
                else if (value == 5)
                {
                        int number = stoi(data);
                        if (this->number_stud == number)
                                cout << *this << endl;
                        return true;
                }
                else if (value == 6)
                {
                        int number = 0;
                        if (data == "true" || data == "true" || data == "1")
                                number = 1;
                        else
                                number = 0;

                        if (this->debt == number)
                                return 1;
                        else
                                return 0;
                }
        }
        catch (const std::exception& ex)
```

```
	{
		cout << ex.what() << endl;
		return 0;
	}

	return 0;
}

ostream& operator<< (ostream& output, const Student& other)
{
	output << other.get_info();
	return output;
}

bool Student::operator==(const int ns) const
{
	return this->number_stud == ns;
}

Student::Student(int a, int n, int m, string na, bool d, int pd) : age(a),
number_stud(n), middle_mark(m), name(na), debt(d), prog_d(pd)
{
	//cout << "\nВызвался конструктор с параметрами";
}
Student::Student() : age(0), number_stud(0), middle_mark(0), name("Name"), debt(0),
prog_d(0)
{
	//cout << "\nВызвался конструктор по умолчанию.";
}
Student::Student(const Student& other) : age(other.age), number_stud(other.number_stud),
middle_mark(other.middle_mark), name(other.name), debt(other.debt), prog_d(other.prog_d)
{
	//cout << "\nВызвался конструктор копирования.";
}
Student::~Student()
{
	//cout << "\nВызвался деструктор";

}
```

class2.cpp

```
#include "class2.h"

stringstream Course::get_str() const
{
	stringstream temp;

	temp << " " << age << " " << number_stud << " " << middle_mark << " "
		<< name << " " << debt << " " << prog_d << " " << course;

	return temp;
}

string Course::get_info() const
{
	stringstream temp;

	temp.setf(ios::left);
	temp << setw(10) << age << setw(8) << number_stud << setw(16) << middle_mark <<
setw(9)
		<< name << setw(7) << debt << setw(14) << prog_d << setw(4) << course;

	return temp.str();
}
```

```
int Course::countElement(int value, string data)
{
	try
	{
		if (value == 1)
		{
			if (this->name == data)
				return 1;
			else
				return 0;
		}
		else if (value == 2)
		{
			int number = stoi(data);
			if (this->age == number)
				return 1;
			else
				return 0;
		}
		else if (value == 3)
		{
			int number = stoi(data);
			if (this->middle_mark == number)
				return 1;
			else
				return 0;
		}
		else if (value == 4)
		{
			int number = stoi(data);
			if (this->prog_d == number)
				return 1;
			else
				return 0;
		}
		else if (value == 5)
		{
			int number = stoi(data);
			if (this->number_stud == number)
				return 1;
			else
				return 0;
		}
		else if (value == 6)
		{
			int number = 0;
			if (data == "true" || data == "true" || data == "1")
				number = 1;
			else
				number = 0;

			if (this->debt == number)
				return 1;
			else
				return 0;
		}
		else if (value == 7)
		{
			int number = stoi(data);
			if (this->course == number)
				return 1;
			else
				return 0;
		}
```

```
	}
	catch (const std::exception & ex)
	{
		cout << ex.what() << endl;
		return 0;
	}

	return 0;
}

bool Course::elementOutput(int value, string data)
{
	try
	{
		if (value == 1)
		{
			if (this->name == data)
				cout << *this << endl;
			return true;
		}
		else if (value == 2)
		{
			int number = stoi(data);
			if (this->age == number)
				cout << *this << endl;
			return true;
		}
		else if (value == 3)
		{
			int number = stoi(data);
			if (this->middle_mark == number)
				cout << *this << endl;
			return true;
		}
		else if (value == 4)
		{
			int number = stoi(data);
			if (this->prog_d == number)
				cout << *this << endl;
			return true;
		}
		else if (value == 5)
		{
			int number = stoi(data);
			if (this->number_stud == number)
				cout << *this << endl;
			return true;
		}
		else if (value == 6)
		{
			int number = 0;
			if (data == "true" || data == "true" || data == "1")
				number = 1;
			else
				number = 0;

			if (this->debt == number)
				return 1;
			else
				return 0;
		}
		else if (value == 7)
		{
			int number = stoi(data);
			if (this->course == number)
```

```
                        cout << *this << endl;
                return true;
            }
        }
        catch (const std::exception & ex)
        {
            cout << ex.what() << endl;
            return 0;
        }

        return 0;
}

Course::Course(int a, int n, int m, string na, bool d, int pd, int c) : Student(a, n, m,
na, d, pd), course(c) {}
Course::Course() : Student(), course(0) {}
Course::Course(const Course& other) : Student(other), course(other.course) {}
Course::~Course() {}

bool Course::operator==(const int ns) const
{
        return this->number_stud == ns;
}
```

main.cpp

```
#include "Header.h"

#include "class1.h"

#include "class2.h"



Student* newProgram(int);

void VectorMenu();

void ListMenu();

void MapMenu();

void SetMenu();



int main()

{

        setlocale(LC_ALL, "ru");
```

```
int choise = 0;
bool stop = 1;

while (stop)
{
	cout << "Выберите STL контейнер:" << endl;
	cout << "1) Vector;" << endl;
	cout << "2) List;" << endl;
	cout << "3) Map;" << endl;
	cout << "4) Set;" << endl;
	cout << "5) Выход;" << endl;
	cout << "======================" << endl;
	cout << "Пункт: ";
	cin >> choise;

	switch (choise)
	{
	case 1:
		VectorMenu();
		break;

	case 2:
		ListMenu();
		break;
```

```
		case 3:
			MapMenu();
			break;

		case 4:
			SetMenu();
			break;

		case 5:
			stop = 0;
			break;

		default:
			cout << "Ошибка. Неверная команда. Повторите попытку." << endl;
		}
}

if (_CrtDumpMemoryLeaks())
		cout << endl << "Обнаружена утечка памяти!" << endl;
else
		cout << endl << "Утечки не обнаруженно!" << endl;

system("PAUSE");
```

```
	return 0;
}

Student* newProgram(int value)
{
	if (value % 2 == 0)
	{
		Student* temp = new Course(17, 4, 10, "Peter", 0, 0, 1);
		return temp;
	}
	else
	{
		Student* temp = new Student(19, 4, 9, "Jhon", 1, 14);
		return temp;
	}
}

void VectorMenu()
{
	vector <unique_ptr<Student>> vector;
	std::vector<unique_ptr<Student>>::iterator it;
	stringstream temp;
	string data;
	bool stop = 1, findEl = 0;
```

```
int choise = 0, choise2 = 0, choise3 = 0;

int value = 0, number = 0, result = 0, sum = 0;



for (size_t i = 0; i < 4; i++)

{

	if (i == 0)

		vector.emplace_back(new Student());

	else if (i == 1)

		vector.emplace_back(new Course(20, 1, 10, "Den", 0, 0, 3));

	else if (i == 2)

		vector.emplace_back(new Student(18, 2, 8, "Dima", 0, 0));

	else if (i == 3)

		vector.emplace_back(new Course(19, 3, 7, "Gordon", 1, 25, 2));

}



while (stop != 0)

{

	if (vector.size() == 0)

	{

		cout << "Вектор пуст. Что вы хотите сделать?" << endl;

		cout << "1) Добавить элемент" << endl;

		cout << "2) Завершение работы" << endl;

		cout << "====================================" << endl;

		cout << "Ваш выбор: ";
```

```
cin >> choise;
cout << endl;

switch (choise)
{
case 1:
	cout << "Выберите программу, которую хотите добавить:" << endl;
	cout << "1. Элемент класса Student" << endl;
	cout << "2. Элемент класса Course" << endl;
	cout << "=========================================" << endl;
	cout << "Ваш выбор: ";
	cin >> value;

	try
	{
		vector.at(value);

		if (value == 1 || value == 2)
		{
			vector.emplace_back(newProgram(value));
			cout << "Элемент добавлен." << endl;
		}
		else
```

```
                                        cout << "Ошибка. Неверный номер." << endl;
                }
                catch (const std::exception& ex)
                {
                        cout << ex.what() << endl;
                }

                break;

        case 2:
                cout << "Завершение работы." << endl;
                stop = 0;
                break;

        default:
                cout << "Неверный номер элемента. Повторите попытку." << endl;
                break;
        }
}
else
{
        cout << endl;
        cout << "1)Вывод на экран" << endl;
        cout << "2)Удаление элемента" << endl;
```

```
        cout << "3)Добавление элементов" << endl;
        cout << "4)Завершение работы" << endl;
        cout << "======================" << endl;
        cout << "Ваш выбор: ";
        cin >> choise;
        cout << endl;
}

switch (choise)
{
case 1:
        cout << "Выберите команду:" << endl;
        cout << "1) Вывести весь список на экран" << endl;
        cout << "2) Вывести студента по номеру" << endl;
        cout << "3) Вывести количество элементов по критерию" << endl;
        cout << "4) Найти элемент по критерию" << endl;
        cout << "5) Вернуться к выбору действий" << endl;
        cout << "====================================" << endl;
        cout << "Ваш выбор: ";
        cin >> choise2;
        cout << endl;

        switch (choise2)
        {
```

```
case 1:
        cout << setw(10) << "Имя" << setw(8) << "Номер";
        cout << setw(15) << "Средний балл" << setw(7) << "Имя";
        cout << setw(10) << "Долг" << setw(14) << "Долг(прог.)";
        cout << setw(7) << "Курс" << endl;

        number = 1;
        for_each(vector.begin(), vector.end(), [&number](const unique_ptr<Student>&
program)
        {
            cout << number << ". " << *program << endl;
            number++;
        });
        number = 1;
        break;

case 2:
        cout << "Введите номер элемента, которого вы хотите получить: ";
        cin >> value;
        cout << endl;

        findEl = 0, number = -1;
        for (const auto& element : vector)
        {
```

```
                if (element->get_numb() == value)
                {
                        number++;
                        findEl = 1;
                        break;
                }
                else
                        number++;
        }

        if (findEl)
        {
                temp = vector[number]->get_str();
                data = temp.str();
                cout << "Ваш элемент: " << endl;
                cout << data << endl << endl;
        }
        else
                cout << "Элемент с таким номером не найден." << endl;

        break;
case 3:
        cout << "Выберите критерий, по которому надо искать: " << endl;
        cout << "1) Имя" << endl;
```

```
cout << "2) Возраст" << endl;

cout << "3) Средний балл" << endl;

cout << "4) Долг по прог." << endl;

cout << "5) Номер" << endl;

cout << "6) Есть ли долг" << endl;

cout << "7) Вернуться назад" << endl;

cout << "====================================" << endl;

cout << "Ваш выбор: ";

cin >> choise3;

cout << endl;


if (choise3 < 1 || choise3 >= 7)

{

	cout << "Возвращение назад." << endl;

	break;

}


it = vector.begin();


cout << "Введите критерий: ";

cin.ignore();

getline(cin, data);

number = 0, value = 0;
```

```
                while (number < vector.size())
                {
                        result = (*it)->countElement(choise3, data);
                        number++;
                        it++;
                        sum += result;
                }
                if (sum != 0)
                        cout << "Количество элементов с данным параметром: " << sum <<
endl;

                break;

        case 4:
                cout << "Выберите критерий, по которому надо искать: " << endl;
                cout << "1) Имя" << endl;
                cout << "2) Возраст" << endl;
                cout << "3) Средний балл" << endl;
                cout << "4) Долг по прог." << endl;
                cout << "5) Номер" << endl;
                cout << "6) Есть ли долг" << endl;
                cout << "7) Вернуться назад" << endl;
                cout << "====================================" << endl;
                cout << "Ваш выбор: ";
```

```
cin >> choise3;
cout << endl;

if (choise3 < 1 || choise3 >= 7)
{
	cout << "Возвращение назад." << endl;
	break;
}

it = vector.begin();
cout << "Введите критерий: ";
cin.ignore();
getline(cin, data);
number = 0, value = 0;

while (number < vector.size())
{
	result = (*it)->elementOutput(choise3, data);
	number++;
	it++;
}

break;
case 5:
```

```
            cout << "Возвращение назад." << endl;
            break;

        default:
            cout << "Неверный символ. Повторите попытку." << endl;
            break;

        }
        break;

case 2:
        cout << "Введите номер элемента, который хотите удалить: ";
        cin >> value;
        cout << endl;

        findEl = 0, number = -1;
        for (const auto& element : vector)
        {
            if (element->get_numb() == value)
            {
                number++;
                findEl = 1;
                break;
            }
```

```
            else
                number++;
        }

        if (findEl)
        {
            it = vector.begin();
            advance(it, number);
            vector.erase(it);

            cout << "Удаление выполнено." << endl;
        }
        else
            cout << "Элемент не найден." << endl;

        break;

case 3:
        cout << "Выберите программу, которую хотите добавить:" << endl;
        cout << "1. Элемент класса Student" << endl;
        cout << "2. Элемент класса Course" << endl;
        cout << "========================================" << endl;
        cout << "Ваш выбор: ";
        cin >> value;
```

```
            try
            {
                vector.at(value);

                if (value == 1 || value == 2)
                {
                    vector.emplace_back(newProgram(value));
                    cout << "Элемент добавлен." << endl;
                }
                else
                    cout << "Ошибка. Неверный номер." << endl;
            }
            catch (const std::exception & ex)
            {
                cout << ex.what() << endl;
            }

            break;

        case 4:
            cout << "Завершение работы." << endl << endl;
            stop = 0;
            break;
```

```
                default:
                        cout << "Неверный символ. Повторите попытку." << endl;
                        break;

                }
        }
}

void ListMenu()
{
        list <unique_ptr<Student>> list;
        stringstream temp;
        string data;
        bool stop = 1, findEl = 0;
        int choise = 0, choise2 = 0, choise3 = 0;
        int value = 0;
        int number = 0;
        int result = 0, sum = 0;
        auto it = list.begin();

        for (size_t i = 0; i < 4; i++)
        {
                if (i == 0)
```

```
                list.emplace_back(new Student());
        else if (i == 1)
                list.emplace_back(new Course(20, 1, 10, "Den", 0, 0, 3));
        else if (i == 2)
                list.emplace_back(new Student(18, 2, 8, "Dima", 0, 0));
        else if (i == 3)
                list.emplace_back(new Course(19, 3, 7, "Gordon", 1, 25, 2));
}

while (stop != 0)
{
        if (list.size() == 0)
        {
                cout << "Вектор пуст. Что вы хотите сделать?" << endl;
                cout << "1) Добавить элемент" << endl;
                cout << "2) Завершение работы" << endl;
                cout << "=====================================" << endl;
                cout << "Ваш выбор: ";
                cin >> choise;
                cout << endl;

                switch (choise)
                {
                case 1:
```

```
cout << "Выберите программу, которую хотите добавить:" << endl;
cout << "1. Элемент класса Student" << endl;
cout << "2. Элемент класса Course" << endl;
cout << "=========================================" << endl;
cout << "Ваш выбор: ";
cin >> value;

try
{
    if (value == 1 || value == 2)
    {
        list.emplace_front(newProgram(value));
        cout << "Элемент добавлен." << endl;
    }
    else
        cout << "Ошибка. Неверный номер." << endl;
}
catch (const std::exception & ex)
{
    cout << ex.what() << endl;
}

break;
```

```
        case 2:
            cout << "Завершение работы." << endl;
            stop = 0;
            break;

        default:
            cout << "Неверный номер элемента. Повторите попытку." << endl;
            break;
        }
}
else
{
        cout << endl;
        cout << "1)Вывод на экран" << endl;
        cout << "2)Удаление элемента" << endl;
        cout << "3)Добавление элементов" << endl;
        cout << "4)Завершение работы" << endl;
        cout << "======================" << endl;
        cout << "Ваш выбор: ";
        cin >> choise;
        cout << endl;
}
```

```
switch (choise)
{
case 1:
	cout << "Выберите команду:" << endl;
	cout << "1) Вывести весь список на экран" << endl;
	cout << "2) Вывести студента по номеру" << endl;
	cout << "3) Вывести количество элементов по критерию" << endl;
	cout << "4) Найти элемент по критерию" << endl;
	cout << "5) Вернуться к выбору действий" << endl;
	cout << "=====================================" << endl;
	cout << "Ваш выбор: ";
	cin >> choise2;
	cout << endl;

	switch (choise2)
	{
	case 1:
		cout << setw(10) << "Имя" << setw(8) << "Номер";
		cout << setw(15) << "Средний балл" << setw(7) << "Имя";
		cout << setw(10) << "Долг" << setw(14) << "Долг(прог.)";
		cout << setw(7) << "Курс" << endl;

		number = 1;
```

```
                for_each(list.begin(), list.end(), [&number](const unique_ptr<Student>&
program)
                {
                        cout << number << ". " << *program << endl;
                        number++;
                });
                number = 1;
                break;

        case 2:
                cout << "Введите номер элемента, которого вы хотите получить: ";
                cin >> value;
                cout << endl;

                findEl = 0, number = -1;
                for (const auto& element : list)
                {
                        if (element->get_numb() == value)
                        {
                                number++;
                                findEl = 1;
                                break;
                        }
                        else
```

```
                    number++;
                }

                if (findEl)
                {
                    it = list.begin();
                    advance(it, number);

                    temp = (*it)->get_str();
                    data = temp.str();

                    cout << "Ваш элемент: " << endl;
                    cout << data << endl << endl;
                }
                else
                    cout << "Элемент с таким номером не найден." << endl;

                break;

            case 3:
                cout << "Выберите критерий, по которому надо искать: " << endl;
                cout << "1) Имя" << endl;
                cout << "2) Возраст" << endl;
                cout << "3) Средний балл" << endl;
```

```
cout << "4) Долг по прог." << endl;

cout << "5) Номер" << endl;

cout << "6) Есть ли долг" << endl;

cout << "7) Вернуться назад" << endl;

cout << "=====================================" << endl;

cout << "Ваш выбор: ";

cin >> choise3;

cout << endl;



if (choise3 < 1 || choise3 >= 7)

{

	cout << "Возвращение назад." << endl;

	break;

}



it = list.begin();

result = 0, sum = 0;

cout << "Введите критерий: ";

cin.ignore();

getline(cin, data);

number = 0, value = 0;



while (number < list.size())

{
```

```
                result = (*it)->countElement(choise3, data);
                number++;
                it++;
                sum += result;
            }
            if (sum != 0)
                cout << "Количество элементов с данным параметром: " << sum <<
endl;

            break;

        case 4:
            cout << "Выберите критерий, по которому надо искать: " << endl;
            cout << "1) Имя" << endl;
            cout << "2) Возраст" << endl;
            cout << "3) Средний балл" << endl;
            cout << "4) Долг по прог." << endl;
            cout << "5) Номер" << endl;
            cout << "6) Есть ли долг" << endl;
            cout << "7) Вернуться назад" << endl;
            cout << "=====================================" << endl;
            cout << "Ваш выбор: ";
            cin >> choise3;
            cout << endl;
```

```
            if (choise3 < 1 || choise3 >= 7)
            {
                cout << "Возвращение назад." << endl;
                break;
            }

            it = list.begin();
            cout << "Введите критерий: ";
            cin.ignore();
            getline(cin, data);
            number = 0, value = 0;

            while (number < list.size())
            {
                result = (*it)->elementOutput(choise3, data);
                number++;
                it++;
            }

            break;
        case 5:
            cout << "Возвращение назад." << endl;
            break;
```

```
        default:
                cout << "Неверный символ. Повторите попытку." << endl;
                break;

        }
        break;

case 2:
        cout << "Введите номер элемента, который хотите удалить: ";
        cin >> value;
        cout << endl;

        findEl = 0, number = -1;
        for (const auto& element : list)
        {
                if (element->get_numb() == value)
                {
                        number++;
                        findEl = 1;
                        break;
                }
                else
                        number++;
```

```
            }

            if (findEl)
            {
                it = list.begin();
                advance(it, number);
                list.erase(it);

                cout << "Удаление выполнено." << endl;
            }
            else
                cout << "Элемент не найден." << endl;

            break;

case 3:
            cout << "Выберите программу, которую хотите добавить:" << endl;
            cout << "1. Элемент класса Student" << endl;
            cout << "2. Элемент класса Course" << endl;
            cout << "=========================================" << endl;
            cout << "Ваш выбор: ";
            cin >> value;

            try
```

```
            {
                if (value == 1 || value == 2)
                {
                    list.emplace_front(newProgram(value));
                    cout << "Элемент добавлен." << endl;
                }
                else
                    cout << "Ошибка. Неверный номер." << endl;
            }
            catch (const std::exception & ex)
            {
                cout << ex.what() << endl;
            }

            break;

        case 4:
            cout << "Завершение работы." << endl << endl;
            stop = 0;
            break;

        default:
            cout << "Неверный символ. Повторите попытку." << endl;
            break;
```

```
                }
        }
}

void MapMenu()
{
        map <int, unique_ptr<Student>> map;
        stringstream temp;
        string data;
        bool stop = 1, findEl = 0;
        int choise = 0, choise2 = 0, choise3 = 0;
        int value = 0;
        int i = 0;
        int number = 0, sum = 0, result = 0;
        auto it = map.begin();

        for (; i < 4; i++)
        {
                if (i == 0)
                        map.emplace(i + 1, new Student());
                else if (i == 1)
                        map.emplace(i + 1, new Course(20, 1, 10, "Den", 0, 0, 3));
                else if (i == 2)
```

```
                map.emplace(i + 1, new Student(18, 2, 8, "Dima", 0, 0));
        else if (i == 3)
                map.emplace(i + 1, new Course(19, 3, 7, "Gordon", 1, 25, 2));
}

while (stop != 0)
{
        if (map.size() == 0)
        {
                cout << "Вектор пуст. Что вы хотите сделать?" << endl;
                cout << "1) Добавить элемент" << endl;
                cout << "2) Завершение работы" << endl;
                cout << "=====================================" << endl;
                cout << "Ваш выбор: ";
                cin >> choise;
                cout << endl;

                switch (choise)
                {
                case 1:
                        cout << "Выберите программу, которую хотите добавить:" << endl;
                        cout << "1. Элемент класса Student" << endl;
                        cout << "2. Элемент класса Course" << endl;
```

```
                cout << "=========================================" <<
endl;

                cout << "Ваш выбор: ";

                cin >> value;


                try

                {

                        if (value == 1 || value == 2)

                        {

                                map.emplace(++i, newProgram(value));

                                cout << "Элемент добавлен." << endl;

                        }

                        else

                                cout << "Ошибка. Неверный номер." << endl;

                }

                catch (const std::exception & ex)

                {

                        cout << ex.what() << endl;

                }


                break;


        case 2:

                cout << "Завершение работы." << endl;
```

```
                stop = 0;
                break;

        default:
                cout << "Неверный номер элемента. Повторите попытку." << endl;
                break;
        }
}
else
{
        cout << endl;
        cout << "1)Вывод на экран" << endl;
        cout << "2)Удаление элемента" << endl;
        cout << "3)Добавление элементов" << endl;
        cout << "4)Завершение работы" << endl;
        cout << "======================" << endl;
        cout << "Ваш выбор: ";
        cin >> choise;
        cout << endl;
}

switch (choise)
{
case 1:
```

```
cout << "Выберите команду:" << endl;
cout << "1) Вывести весь список на экран" << endl;
cout << "2) Вывести студента по номеру" << endl;
cout << "3) Вывести количество элементов по критерию" << endl;
cout << "4) Найти элемент по критерию" << endl;
cout << "5) Вернуться к выбору действий" << endl;
cout << "====================================" << endl;
cout << "Ваш выбор: ";
cin >> choise2;
cout << endl;

switch (choise2)
{
case 1:
	cout << setw(10) << "Имя" << setw(8) << "Номер";
	cout << setw(15) << "Средний балл" << setw(7) << "Имя";
	cout << setw(10) << "Долг" << setw(14) << "Долг(прог.)";
	cout << setw(7) << "Курс" << endl;

	for_each(map.begin(), map.end(), [](const std::pair<const int,
unique_ptr<Student>>& program)
	{
		cout << program.first << ". " << *program.second << endl;
	});
```

```
            break;


case 2:
            cout << "Введите номер элемента, которого вы хотите получить: ";
            cin >> value;
            cout << endl;


            findEl = 0;
            it = map.find(value);


            if (it != map.end())
            {
                        temp = (*it).second->get_str();
                        data = temp.str();


                        cout << "Ваш элемент: " << endl;
                        cout << data << endl << endl;
            }
            else
                        cout << "Элемент с таким номером не найден." << endl;


            break;
```

```
case 3:
		cout << "Выберите критерий, по которому надо искать: " << endl;
		cout << "1) Имя" << endl;
		cout << "2) Возраст" << endl;
		cout << "3) Средний балл" << endl;
		cout << "4) Долг по прог." << endl;
		cout << "5) Номер" << endl;
		cout << "6) Есть ли долг" << endl;
		cout << "7) Вернуться назад" << endl;
		cout << "=====================================" << endl;
		cout << "Ваш выбор: ";
		cin >> choise3;
		cout << endl;

		if (choise3 < 1 || choise3 >= 7)
		{
			cout << "Возвращение назад." << endl;
			break;
		}

		it = map.begin();
		result = 0, sum = 0;
		cout << "Введите критерий: ";
		cin.ignore();
```

```
                getline(cin, data);
                number = 0, value = 0;

                while (number < map.size())
                {
                        result = it->second->countElement(choise3, data);
                        number++;
                        it++;
                        sum += result;
                }
                if (sum != 0)
                        cout << "Количество элементов с данным параметром: " << sum <<
endl;

                break;

        case 4:
                cout << "Выберите критерий, по которому надо искать: " << endl;
                cout << "1) Имя" << endl;
                cout << "2) Возраст" << endl;
                cout << "3) Средний балл" << endl;
                cout << "4) Долг по прог." << endl;
                cout << "5) Номер" << endl;
                cout << "6) Есть ли долг" << endl;
```

```
cout << "7) Вернуться назад" << endl;
cout << "=====================================" << endl;
cout << "Ваш выбор: ";
cin >> choise3;
cout << endl;

if (choise3 < 1 || choise3 >= 7)
{
	cout << "Возвращение назад." << endl;
	break;
}

it = map.begin();
cout << "Введите критерий: ";
cin.ignore();
getline(cin, data);
number = 0, value = 0;

while (number < map.size())
{
	result = it->second->elementOutput(choise3, data);
	number++;
	it++;
}
```

```
            break;
        case 5:
            cout << "Возвращение назад." << endl;
            break;

        default:
            cout << "Неверный символ. Повторите попытку." << endl;
            break;

        }
        break;

case 2:
        cout << "Введите номер элемента, который хотите удалить: ";
        cin >> value;
        cout << endl;

        findEl = 0;
        it = map.find(value);

        if (it != map.end())
        {
            map.erase(it);
```

```
            cout << "Удаление выполнено." << endl;
        }
        else
            cout << "Элемент не найден." << endl;

        break;

case 3:
        cout << "Выберите программу, которую хотите добавить:" << endl;
        cout << "1. Элемент класса Student" << endl;
        cout << "2. Элемент класса Course" << endl;
        cout << "=========================================" << endl;
        cout << "Ваш выбор: ";
        cin >> value;

        try
        {
            if (value == 1 || value == 2)
            {
                map.emplace(++i, newProgram(value));
                cout << "Элемент добавлен." << endl;
            }
            else
                cout << "Ошибка. Неверный номер." << endl;
```

```
                }
                catch (const std::exception & ex)
                {
                        cout << ex.what() << endl;
                }

                break;

        case 4:
                cout << "Завершение работы." << endl << endl;
                stop = 0;
                break;

        default:
                cout << "Неверный символ. Повторите попытку." << endl;
                break;

        }
    }
}

void SetMenu()
{
    set <unique_ptr<Student>> set;
```

```
stringstream temp;

string data;

bool stop = 1, findEl = 0;

int choise = 0, choise2 = 0, choise3 = 0;

int value = 0, number = 0, result = 0, sum = 0;

auto it = set.begin();



for (size_t i = 0; i < 4; i++)

{

	if (i == 0)

		set.emplace(new Student());

	else if (i == 1)

		set.emplace(new Course(20, 1, 10, "Den", 0, 0, 3));

	else if (i == 2)

		set.emplace(new Student(18, 2, 8, "Dima", 0, 0));

	else if (i == 3)

		set.emplace(new Course(19, 3, 7, "Gordon", 1, 25, 2));

}



while (stop != 0)

{

	if (set.size() == 0)

	{

		cout << "Вектор пуст. Что вы хотите сделать?" << endl;
```

```
cout << "1) Добавить элемент" << endl;

cout << "2) Завершение работы" << endl;

cout << "=====================================" << endl;

cout << "Ваш выбор: ";

cin >> choise;

cout << endl;



switch (choise)

{

case 1:

	cout << "Выберите программу, которую хотите добавить:" << endl;

	cout << "1. Элемент класса Student" << endl;

	cout << "2. Элемент класса Course" << endl;

	cout << "=========================================" <<
endl;

	cout << "Ваш выбор: ";

	cin >> value;



	try

	{

		if (value == 1 || value == 2)

		{

			set.emplace(newProgram(value));

			cout << "Элемент добавлен." << endl;
```

```
                }
                else
                    cout << "Ошибка. Неверный номер." << endl;
            }
            catch (const std::exception & ex)
            {
                cout << ex.what() << endl;
            }

            break;

        case 2:
            cout << "Завершение работы." << endl;
            stop = 0;
            break;

        default:
            cout << "Неверный номер элемента. Повторите попытку." << endl;
            break;
        }
    }
    else
    {
        cout << endl;
```

```
        cout << "1)Вывод на экран" << endl;
        cout << "2)Удаление элемента" << endl;
        cout << "3)Добавление элементов" << endl;
        cout << "4)Завершение работы" << endl;
        cout << "======================" << endl;
        cout << "Ваш выбор: ";
        cin >> choise;
        cout << endl;
}

switch (choise)
{
case 1:
        cout << "Выберите команду:" << endl;
        cout << "1) Вывести весь список на экран" << endl;
        cout << "2) Вывести студента по номеру" << endl;
        cout << "3) Вывести количество элементов по критерию" << endl;
        cout << "4) Найти элемент по критерию" << endl;
        cout << "5) Вернуться к выбору действий" << endl;
        cout << "====================================" << endl;
        cout << "Ваш выбор: ";
        cin >> choise2;
        cout << endl;
```

```
switch (choise2)
{
case 1:
	cout << setw(10) << "Имя" << setw(8) << "Номер";
	cout << setw(15) << "Средний балл" << setw(7) << "Имя";
	cout << setw(10) << "Долг" << setw(14) << "Долг(прог.)";
	cout << setw(7) << "Курс" << endl;

	number = 1;
	for_each(set.begin(), set.end(), [&number](const unique_ptr<Student>&
program)
	{
		cout << number << ". " << *program << endl;
		number++;
	});
	number = 1;
	break;

case 2:
	cout << "Введите номер элемента, которого вы хотите получить: ";
	cin >> value;
	cout << endl;

	findEl = 0, number = -1;
```

```
for (const auto& element : set)
{
	if (element->get_numb() == value)
	{
		number++;
		findEl = 1;
		break;
	}
	else
		number++;
}

if (findEl)
{
	it = set.begin();
	advance(it, number);

	temp = (*it)->get_str();
	data = temp.str();

	cout << "Ваш элемент: " << endl;
	cout << data << endl << endl;
}
else
```

```
                cout << "Элемент с таким номером не найден." << endl;


            break;


        case 3:

            cout << "Выберите критерий, по которому надо искать: " << endl;

            cout << "1) Имя" << endl;

            cout << "2) Возраст" << endl;

            cout << "3) Средний балл" << endl;

            cout << "4) Долг по прог." << endl;

            cout << "5) Номер" << endl;

            cout << "6) Есть ли долг" << endl;

            cout << "7) Вернуться назад" << endl;

            cout << "=====================================" << endl;

            cout << "Ваш выбор: ";

            cin >> choise3;

            cout << endl;


            if (choise3 < 1 || choise3 >= 7)

            {

                cout << "Возвращение назад." << endl;

                break;

            }
```

```
                it = set.begin();
                result = 0, sum = 0;
                cout << "Введите критерий: ";
                cin.ignore();
                getline(cin, data);
                number = 0, value = 0;

                while (number < set.size())
                {
                        result = (*it)->countElement(choise3, data);
                        number++;
                        it++;
                        sum += result;
                }
                if (sum != 0)
                        cout << "Количество элементов с данным параметром: " << sum <<
endl;

                break;

        case 4:
                cout << "Выберите критерий, по которому надо искать: " << endl;
                cout << "1) Имя" << endl;
                cout << "2) Возраст" << endl;
```

```
cout << "3) Средний балл" << endl;

cout << "4) Долг по прог." << endl;

cout << "5) Номер" << endl;

cout << "6) Есть ли долг" << endl;

cout << "7) Вернуться назад" << endl;

cout << "====================================" << endl;

cout << "Ваш выбор: ";

cin >> choise3;

cout << endl;



if (choise3 < 1 || choise3 >= 7)

{

	cout << "Возвращение назад." << endl;

	break;

}



it = set.begin();

cout << "Введите критерий: ";

cin.ignore();

getline(cin, data);

number = 0, value = 0;



while (number < set.size())

{
```

```
                result = (*it)->elementOutput(choise3, data);
                number++;
                it++;
            }

            break;
        case 5:
            cout << "Возвращение назад." << endl;
            break;

        default:
            cout << "Неверный символ. Повторите попытку." << endl;
            break;

        }
        break;

case 2:
        cout << "Введите номер элемента, который хотите удалить: ";
        cin >> value;
        cout << endl;

        findEl = 0, number = -1;
        for (const auto& element : set)
```

```
{
	if (element->get_numb() == value)
	{
		number++;
		findEl = 1;
		break;
	}
	else
		number++;
}

if (findEl)
{
	it = set.begin();
	advance(it, number);
	set.erase(it);

	cout << "Удаление выполнено." << endl;
}
else
	cout << "Элемент не найден." << endl;

break;
```

```
case 3:
        cout << "Выберите программу, которую хотите добавить:" << endl;
        cout << "1. Элемент класса Student" << endl;
        cout << "2. Элемент класса Course" << endl;
        cout << "=========================================" << endl;
        cout << "Ваш выбор: ";
        cin >> value;

        try
        {
            if (value == 1 || value == 2)
            {
                set.emplace(newProgram(value));
                cout << "Элемент добавлен." << endl;
            }
            else
                cout << "Ошибка. Неверный номер." << endl;
        }
        catch (const std::exception & ex)
        {
            cout << ex.what() << endl;
        }

        break;
```

```
                case 4:
                        cout << "Завершение работы." << endl << endl;
                        stop = 0;
                        break;

                default:
                        cout << "Неверный символ. Повторите попытку." << endl;
                        break;

                }
        }
}
```

test.cpp

```
#include "Header.h"

void VectorTest();
void ListTest();
void MapTest();
void SetTest();

int main()
{
        setlocale(LC_ALL, "ru");

        VectorTest();
        ListTest();
        MapTest();
        SetTest();

        if (_CrtDumpMemoryLeaks())
                cout << endl << "Обнаружена утечка памяти!" << endl;
        else
                cout << endl << "Утечки не обнаруженно!" << endl;

        system("PAUSE");
        return 0;
}
void VectorTest()
{
        vector<int> vector = { 1, -5, 20, 555, 0 };
```

```
	int vectorSize = vector.size();
	int newVectorSize;
	int value;
	std::vector<int>::iterator it;
	cout << "Vector" << endl;

	vector.push_back(155);
	newVectorSize = vector.size();
	if (vectorSize != newVectorSize && vector[newVectorSize - 1] == 155)
		cout << "Тест добавления элемента\tвыполнен успешно.\n";
	else
		cout << "Тест добавления элемента\tне выполнен успешно.\n";

	it = vector.begin();
	value = vector[2];
	vector.erase(it + 2);
	newVectorSize = vector.size();
	if (vectorSize == newVectorSize && vector[2] != value)
		cout << "Тест удаления элемента\t\tвыполнен успешно.\n";
	else
		cout << "Тест удаления элемента\t\tне выполнен успешно.\n";

	if (vector[0] == 1)
		cout << "Тест получения элемента\t\tвыполнен успешно.\n";
	else
		cout << "Тест получения элемента\t\tне выполнен успешно.\n";
}
void ListTest()
{
	list<int> list = { 1, -5, 20, 555, 0 };
	int listSize = list.size();
	int value;
	std::list<int>::iterator it;
	std::list<int>::iterator it2;
	cout << endl << "List" << endl;

	list.push_back(155);
	list.push_front(228);
	it = list.begin();
	it2 = list.begin();
	std::advance(it2, list.size() - 1);
	if (listSize != list.size() && *it == 228 && *it2 == 155)
		cout << "Тест добавления элемента\tвыполнен успешно.\n";
	else
		cout << "Тест добавления элемента\tне выполнен успешно.\n";

	it2 = list.begin();
	std::advance(it2, 2);
	list.erase(it2);
	it = list.begin();
	std::advance(it, 2);
	if (list.size() == listSize + 1 && it != it2)
		cout << "Тест удаления элемента\t\tвыполнен успешно.\n";
	else
		cout << "Тест удаления элемента\t\tне выполнен успешно.\n";

	if (*it == 20)
		cout << "Тест получения элемента\t\tвыполнен успешно.\n";
	else
		cout << "Тест получения элемента\t\tне выполнен успешно.\n";
}
void SetTest()
{
	set<int> set = { 1, -5, 20, 555, 0 };
	int setSize = set.size();
```

```
	int value;
	std::set<int>::iterator it;
	std::set<int>::iterator it2;
	cout << endl << "Set" << endl;

	set.insert(155);
	it2 = set.begin();
	std::advance(it2, 4);
	if (setSize != set.size() && *it2 == 155)
		cout << "Тест добавления элемента\tвыполнен успешно.\n";
	else
		cout << "Тест добавления элемента\tне выполнен успешно.\n";

	it2 = set.begin();
	set.erase(it2);
	it = set.begin();
	if (set.size() == setSize && it != it2 && *it == 0)
		cout << "Тест удаления элемента\t\tвыполнен успешно.\n";
	else
		cout << "Тест удаления элемента\t\tне выполнен успешно.\n";

	if (*it == 0)
		cout << "Тест получения элемента\t\tвыполнен успешно.\n";
	else
		cout << "Тест получения элемента\t\tне выполнен успешно.\n";
}
void MapTest()
{
	map <int, int> map = { {1, 1}, {-5,2}, {20, 3}, {555, 4}, {0, 5} };
	int mapSize = map.size();
	std::map<int, int>::iterator it;
	std::map<int, int>::iterator it2;
	cout << endl << "Map" << endl;

	map.insert(std::pair<int, int>(155, 6));
	if (mapSize < map.size())
		cout << "Тест добавления элемента\tвыполнен успешно.\n";
	else
		cout << "Тест добавления элемента\tне выполнен успешно.\n";

	it = map.begin();
	map.erase(it);
	if (mapSize == map.size())
		cout << "Тест удаления элемента\t\tвыполнен успешно.\n";
	else
		cout << "Тест удаления элемента\t\tне выполнен успешно.\n";

	it = map.begin();
	if (map.find(0) == it)
		cout << "Тест получения элемента\t\tвыполнен успешно.\n";
	else
		cout << "Тест получения элемента\t\tне выполнен успешно.\n";
}
```

4 Результати тестування

```
Выберите STL контейнер:
1) Vector;
2) List;
3) Map;
4) Set;
5) Выход;
======================
Пункт: 1

1)Вывод на экран
2)Удаление элемента
3)Добавление элементов
4)Завершение работы
======================
Ваш выбор: 1

Выберите команду:
1) Вывести весь список на экран
2) Вывести студента по номеру
3) Вывести количество элементов по критерию
4) Найти элемент по критерию
5) Вернуться к выбору действий
====================================
Ваш выбор: 1

       Имя   Номер   Средний балл    Имя      Долг   Долг(прог.)   Курс
1. 0          0       0              Name     0      0
2. 20         1       10             Den      0      0             3
3. 18         2       8              Dima     0      0
4. 19         3       7              Gordon   1      25            2

1)Вывод на экран
2)Удаление элемента
3)Добавление элементов
4)Завершение работы
======================
Ваш выбор:
```

5 Опис результатів

Порівняли STL-алгоритми, що не модифікують послідовність. Було створенно меню за варіантами вибору контейнеру. Реалізовані методи роботи з контейнером.