



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика, искусственный интеллект и системы управления» (ИУ)
КАФЕДРА «Информационная безопасность» (ИУ8)

Лабораторная работа №5
ПО КУРСУ
«Изучение использования объектов своих
классов в упорядоченных и неупорядоченных контейнерах
библиотеки STL (set и map, unordered_set и unordered_map)»

Студент

ИУ8-21
(Группа)

Г. А. Карев
(И. О. Фамилия)

Преподаватель:

В. В. Соборова
(И.О. Фамилия)

Цель работы:

Цель работы состоит в овладении навыком использования объектов классов в последовательных контейнерах. Написать рабочую программу для выполнения задачи.

Вариант 12

Задача:

Для класса, разработанного в ЛР4, обеспечить возможность добавления объектов в контейнер `set` (сортировка как указано в задании на ЛР4) и в контейнер `unordered_set`. Исходные данные как в ЛР4 читать из файла, вывести на печать для контроля объекты

Параметры приложений		Исходный контейнер <code>vector</code> , копируем в <code>deque</code>	Исходный контейнер <code>list</code> , копируем в <code>vector</code>
Объект- сотрудник (поля: ФИО, дата приема на работу, должность, базовый оклад)	Сортировка по ФИО	1	34
	Сортировка по окладу	9	4

Объект- банковский вклад (поля: название, сумма вклада, тип валюты, ставка в % годовых)	Сортировка по сумме вклада	5	2
	Сортировка по названию	7	31
Объект- студент (поля: ФИО, группа, номер зачетной книжки, массив 4-х оценок за сессию)	Сортировка по ФИО	3	10
	Сортировка по среднему баллу	11	12

Код:

Header.h (заголовочный файл)

```
#pragma once
#include <string>
#include <functional>
using namespace std;

class Student {
    string name;
    string group;
    string number;
    int grades[4];

public:
    Student();
    Student(const string& name, const string& grp, const string& num, const int g[4]);
    Student(const Student& copy);
    Student(Student&& cmove) noexcept;
    Student& operator=(const Student& givecopy);
```

```

Student& operator=(Student&& givemove) noexcept;
double getGrade() const;
friend ostream& operator<<(ostream& outS, const Student& student);
bool operator<(const Student& other) const;
bool operator==(const Student& other) const;
friend struct hash<Student>;
};

namespace std {
    template<>
    struct hash<Student> {
        size_t operator()(const Student& s) const {
            hash<string> hash_s;
            size_t h1 = hash_s(s.name);
            size_t h2 = hash_s(s.group);
            size_t h3 = hash_s(s.number);
            int sum_grades = s.grades[0] + s.grades[1] + s.grades[2] + s.grades[3];
            return h1 ^ (h2 << 1) ^ (h3 << 2) ^ (sum_grades << 3);
        }
    };
}

```

Header.cpp (файл реализации)

```

#include "Header.h"
#include <numeric>
#include <iostream>

Student::Student() : name(""), group(""), number(""), grades{ 0, 0, 0, 0 } {}

Student::Student(const string& name, const string& grp, const string& num, const int
g[4]) : name(name), group(grp), number(num) {
    for (int i = 0; i < 4; ++i) { grades[i] = g[i]; }
}

Student::Student(const Student& copy) : name(copy.name), group(copy.group),
number(copy.number) {
    for (int i = 0; i < 4; ++i) {
        grades[i] = copy.grades[i];
    }
}

Student::Student(Student&& cmove) noexcept : name(move(cmove.name)),
group(move(cmove.group)), number(move(cmove.number)) {
    for (int i = 0; i < 4; ++i) {
        grades[i] = cmove.grades[i];
    }
}

Student& Student::operator=(const Student& givecopy) {
    if (this != &givecopy) {
        name = givecopy.name;
        group = givecopy.group;
        number = givecopy.number;
        for (int i = 0; i < 4; ++i) { grades[i] = givecopy.grades[i]; }
    }
    return *this;
}

Student& Student::operator=(Student&& givemove) noexcept {
    if (this != &givemove) {
        name = move(givemove.name);

```

```

        group = move(givemove.group);
        number = move(givemove.number);
        for (int i = 0; i < 4; ++i) { grades[i] = givemove.grades[i]; }
    }
    return *this;
}

double Student::getGrade() const {
    return accumulate(begin(grades), end(grades), 0.0) / 4.0;
}

ostream& operator<<(ostream& outS, const Student& student) {
    outS << "ФИО: " << student.name << ", группа: " << student.group << ", номер
зачетки: " << student.number << ", оценки: ";
    for (int grade : student.grades) { outS << grade << " "; }
    outS << ", ср. балл: " << student.getGrade();
    return outS;
}

bool Student::operator<(const Student& other) const {
    return getGrade() < other.getGrade();
}

bool Student::operator==(const Student& other) const {
    return name == other.name && group == other.group && number == other.number &&
grades[0] == other.grades[0]
        && grades[1] == other.grades[1] && grades[2] == other.grades[2] && grades[3]
== other.grades[3];
}

```

zxc.cpp (основной файл)

```

#include <iostream>
#include <fstream>
#include <string>
#include <set>
#include <unordered_set>
#include <algorithm>
#include "Header.h"
using namespace std;

int main() {
    list<Student> studentsList;
    set<Student> studentsSet;
    unordered_set<Student> studentsUnSet;

    ifstream inF("input.txt");
    ofstream outF("output.txt");
    if (inF.is_open()) {
        string name, group, recordNum;
        int grades[4];

        while (inF >> name >> group >> recordNum >> grades[0] >> grades[1] >>
grades[2] >> grades[3]) {
            Student student(name, group, recordNum, grades);
            studentsList.emplace_back(name, group, recordNum, grades);
        }
        inF.close();
    }
    else {
        cerr << "шкибиди файл" << endl;
        return 1;
    }
}

```

```

    }

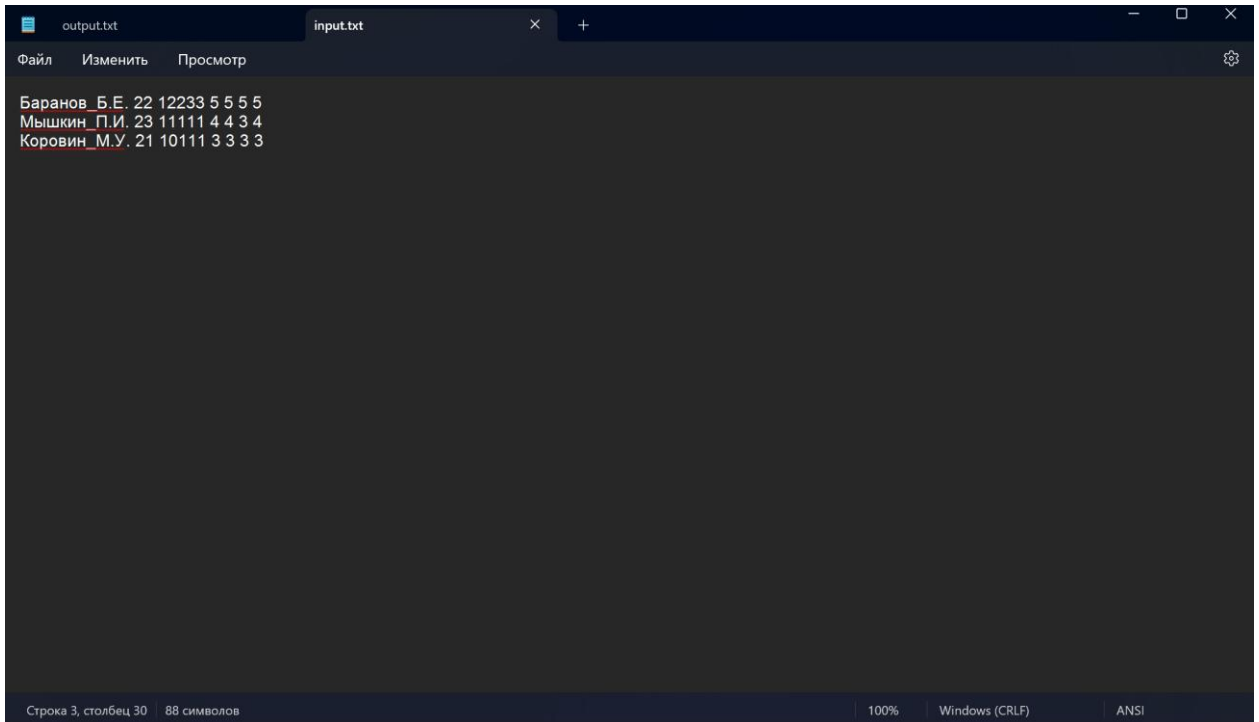
    outF << "Список студентов (list):" << endl;
    for (const auto& student : studentsList) {
        outF << student << endl;
    }
    outF << endl;

    copy(studentsList.begin(), studentsList.end(), inserter(studentsSet,
studentsSet.begin()));
    outF << "Список студентов (set):" << endl;
    for (const auto& student : studentsSet) {
        outF << student << endl;
    }
    outF << endl;

    copy(studentsList.begin(), studentsList.end(), inserter(studentsUnSet,
studentsUnSet.begin()));
    outF << "Список студентов (unordered_set):" << endl;
    for (const auto& student : studentsSet) {
        outF << student << endl;
    }
    outF << endl;
    outF.close();
    return 0;
}

```

Файл для ввода:



The screenshot shows a code editor window with two tabs: 'output.txt' and 'input.txt'. The 'input.txt' tab is active, displaying the following text:

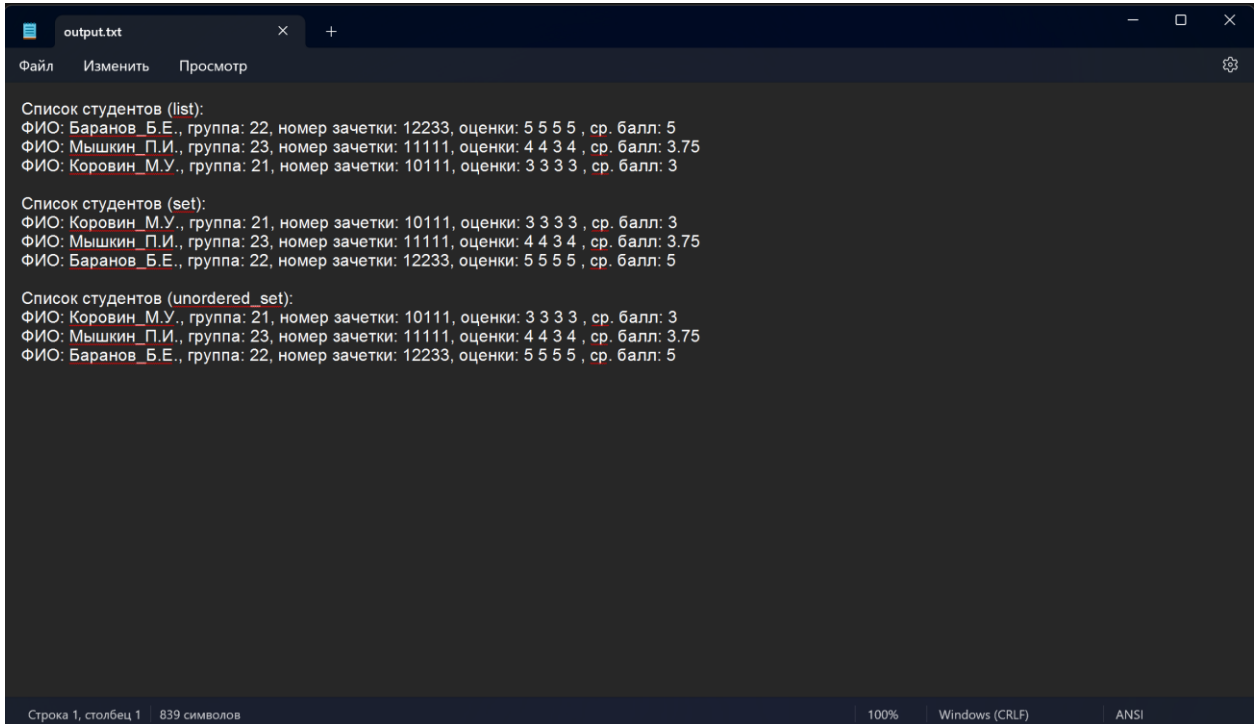
```

Баранов_Б.Е. 22 12233 5 5 5 5
Мышкин_П.И. 23 11111 4 4 3 4
Коровин_М.У. 21 10111 3 3 3 3

```

The editor interface includes a menu bar with 'Файл', 'Изменить', and 'Просмотр'. The status bar at the bottom indicates 'Строка 3, столбец 30 | 88 символов', '100%', 'Windows (CRLF)', and 'ANSI'.

Результаты вывода программы:

A screenshot of a terminal window titled 'output.txt'. The window has a dark background and a light-colored text. The text is organized into three sections: 'Список студентов (list):', 'Список студентов (set):', and 'Список студентов (unordered_set):'. Each section contains three lines of student data, including their full name (ФИО), group number, record number, scores, and average grade. The names are underlined in the original image. The terminal window has a menu bar with 'Файл', 'Изменить', and 'Просмотр'. At the bottom, there is a status bar showing 'Строка 1, столбец 1', '839 символов', '100%', 'Windows (CRLF)', and 'ANSI'.

```
Список студентов (list):
ФИО: Баранов_Б.Е., группа: 22, номер зачетки: 12233, оценки: 5 5 5 5 , ср. балл: 5
ФИО: Мышкин_П.И., группа: 23, номер зачетки: 11111, оценки: 4 4 3 4 , ср. балл: 3.75
ФИО: Коровин_М.У., группа: 21, номер зачетки: 10111, оценки: 3 3 3 3 , ср. балл: 3

Список студентов (set):
ФИО: Коровин_М.У., группа: 21, номер зачетки: 10111, оценки: 3 3 3 3 , ср. балл: 3
ФИО: Мышкин_П.И., группа: 23, номер зачетки: 11111, оценки: 4 4 3 4 , ср. балл: 3.75
ФИО: Баранов_Б.Е., группа: 22, номер зачетки: 12233, оценки: 5 5 5 5 , ср. балл: 5

Список студентов (unordered_set):
ФИО: Коровин_М.У., группа: 21, номер зачетки: 10111, оценки: 3 3 3 3 , ср. балл: 3
ФИО: Мышкин_П.И., группа: 23, номер зачетки: 11111, оценки: 4 4 3 4 , ср. балл: 3.75
ФИО: Баранов_Б.Е., группа: 22, номер зачетки: 12233, оценки: 5 5 5 5 , ср. балл: 5
```

Вывод:

Научился использовать объекты класса в различных контейнерах.
Написал работоспособную программу.