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 (ИУ) «Информатика, искусственный интеллект и системы управления»

Кафедра Информационная безопасность (ИУ8)

Лабораторная работа № 3

ПО КУРСУ

«Алгоритмические языки»

на тему «Изучение перегрузки стандартных операций в языке C++

Студент ИУ8-24 (Группа) А. А. Урнышева (И. О. Фамилия)

Преподаватель: Д. В. Барыкин (И.О. Фамилия)

Цель работы: работа с set и unordered_set в классах

Условие задачи: 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 контейнеров.

Вариант 24:

Текст программы с комментариями:

Файл Header.h

```
#pragma once
#include <iostream>
#include <string>

using namespace std;

class Kredit {
    string name; //название кредита
    int sum; //сумма кредита
    string type_valuta; //тип валюты
    int stavka; //ставка в %

public:
    Kredit(string name, int sum, string type_valuta, int stavka) { //конструктор
        this->name = name;
        this->sum = sum;
        this->type_valuta = type_valuta;
        this->stavka = stavka;
    }

    Kredit(const Kredit& kredit) { //конструктор копирования
        this->name = kredit.name;
        this->sum = kredit.sum;
        this->type_valuta = kredit.type_valuta;
        this->stavka = kredit.stavka;
    }

    Kredit() { //конструктор с нулями
        name = "";
        sum = 0;
        type_valuta = "";
        stavka = 0;
    }

    //дружественные функции для перегрузки операторов ввода-вывода
    friend ostream& operator<<(ostream& os, const Kredit& k1);
    friend istream& operator >>(istream& is, Kredit&);

    //дружественные функции для перегрузки операторов сравнения
    friend bool operator>(const Kredit& k1, const Kredit& k2);
    friend bool operator<(const Kredit& k1, const Kredit& k2);
    friend bool operator ==(const Kredit& k1, const Kredit& k2);

    int getStavka() const {
        return stavka;
```

```
    }
};

ostream& operator<<(ostream& os, const Kredit& k1) { //перегрузка оператора вывода
    os << "Name of kredit: " << k1.name << endl;
    os << "Sum of kredit: " << k1.sum << endl;
    os << "Type of valuta: " << k1.type_valuta << endl;
    os << "Stavka: " << k1.stavka << "%" << endl;
    return os;
}

istream& operator>>(istream& is, Kredit& k1) { //перегрузка оператора ввода
    is >> k1.name;
    is >> k1.sum;
    is >> k1.type_valuta;
    is >> k1.stavka;
    return is;
}

bool operator >(const Kredit& k1, const Kredit& k2) { //перегрузка оператора сравнения в одну сторону
    return k1.stavka > k2.stavka;
}

bool operator <(const Kredit& k1, const Kredit& k2) { //перегрузка оператора сравнения в другую сторону
    return k1.stavka < k2.stavka;
}

bool operator ==(const Kredit& k1, const Kredit& k2) { //перегрузка оператора сравнения в другую сторону
    return k1.stavka == k2.stavka;
}
```

Файл main.cpp

```
#include "Header.h"

#include <vector>
#include <list>
#include <algorithm>
#include <fstream>
#include <set>
#include <unordered_set>

void SaveToFile(const string& filename, const Kredit& k1) { //функция сохранения данных типа Kredit в файл
    ofstream file(filename, ios::app);
    if (file.is_open()) {
        file << k1 << endl;
        cout << k1 << endl;
        file.close();
    }
    else cerr << "Can't open this file :(" << endl;
}

void SaveToFile(const string& filename, const string& s) { //перегрузка функции для сохранения в файл данных типа
string
    ofstream file(filename, ios::app);
    if (file.is_open()) {
        file << s << endl;
        cout << s << endl;
        file.close();
    }
    else cerr << "Can't open this file :(" << endl;
}

struct KreditHash {
    size_t operator()(const Kredit& kredit) const {
        return hash<int>() (kredit.getStavka());
```

```
    }
};

int main() {
    ofstream clear("output.txt");
    clear.close();
    ifstream input("input.txt");

    int n;
    list<Kredit>K1; //создание двусвязного списка
    vector<Kredit>K2; //создание вектора
    set<Kredit>set1; //создание множества
    unordered_set<Kredit, KreditHash> uset1; //создание неупорядоченного множества
    cout << "Enter amount of kredits: ";
    cin >> n;
    cout << endl;

    for (size_t i = 0; i < n; ++i) {
        Kredit k;
        input >> k;
        K1.push_back(k); //добавление элементов из файла в двусвязный список
        set1.insert(k); //добавление элементов из файла в множество
        uset1.insert(k); //добавление элементов из файла в неупорядоченное множество
    }

    string s = "Before sort: ";
    SaveToFile("output.txt", s); //сохранение строки в файл

    for (const auto e : K1) SaveToFile("output.txt", e); //сохранение элементов двусвязного списка в файл

    s = "After sort";

    K1.sort(); //сортировка двусвязного списка

    SaveToFile("output.txt", s); //сохранение строки в файл
    for (const auto e : K1) SaveToFile("output.txt", e); //сохранение отсортированного двусвязного списка в файл

    s = "Vector from list: ";
    SaveToFile("output.txt", s); //сохранение строки в файл


    copy(K1.begin(), K1.end(), back_inserter(K2)); //копирование двусвязного списка в вектор

    for (const auto e : K2) SaveToFile("output.txt", e); //сохранение вектора в файл

    s = "Elements from set: ";
    SaveToFile("output.txt", s);

    for (const auto e : set1) SaveToFile("output.txt", e);

    s = "Elements from unordered set: ";
    SaveToFile("output.txt", s);

    for (const auto e : uset1) SaveToFile("output.txt", e);

    return 0;
        }
```

Файл input.txt

```
name100

1000

type1
```

10

name2

2000

type2

20

name3

3000

type3

5

Файл output.txt

Before sort:

Name of kredit: name100

Sum of kredit: 1000

Type of valuta: type1

Stavka: 10%

Name of kredit: name2

Sum of kredit: 2000

Type of valuta: type2

Stavka: 20%

Name of kredit: name3

Sum of kredit: 3000

Type of valuta: type3

Stavka: 5%

After sort

Name of kredit: name3

Sum of kredit: 3000

Type of valuta: type3

Stavka: 5%

Name of kredit: name100

Sum of kredit: 1000

Type of valuta: type1

Stavka: 10%

Name of kredit: name2

Sum of kredit: 2000

Type of valuta: type2

Stavka: 20%

Vector from list:

Name of kredit: name3

Sum of kredit: 3000

Type of valuta: type3

Stavka: 5%

Name of kredit: name100

Sum of kredit: 1000

Type of valuta: type1

Stavka: 10%

Name of kredit: name2

Sum of kredit: 2000

Type of valuta: type2

Stavka: 20%

Elements from set:

Name of kredit: name3

Sum of kredit: 3000

Type of valuta: type3

Stavka: 5%

Name of kredit: name100

Sum of kredit: 1000

Type of valuta: type1

Stavka: 10%

Name of kredit: name2

Sum of kredit: 2000

Type of valuta: type2

Stavka: 20%

Elements from unordered set:

Name of kredit: name100

Sum of kredit: 1000

Type of valuta: type1

Stavka: 10%

Name of kredit: name2

Sum of kredit: 2000

Type of valuta: type2

Stavka: 20%

Name of kredit: name3

Sum of kredit: 3000

Type of valuta: type3

Stavka: 5%

Вывод: Я добавила элементы из файла в set и unordered_set, программа работает.