ФАКУЛЬТЕТ «Информатика и системы управления»  
КАФЕДРА «Информационная безопасность»

ОТЧЕТ  
по лабораторной работе № 5  
по учебной дисциплине «Алгоритмические языки»  
на тему: «Изучение возможностей наследования классов в языке Си++»

Вариант 23

Выполнил:   
Студент 1 курса, гр. ИУ8-24  
Спиридонов Олег

**Цель работы:**

Изучить принципы использования объектов своих классов в упорядоченных и неупорядоченных контейнерах библиотеки STL (set и map, unordered\_set и unordered\_map).

**Условие задачи:**

Вариант № 23

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.

Файл **input.txt**:

Oleg 5000 ruble 5

Egor 3000 dollar 3

Vlad 4000 euro 4

Uzbek 2000 ruble 2

**Текст программы:**

#include <iostream>

#include <fstream>

#include <sstream>

#include <algorithm>

#include <string>

#include <vector>

#include <set>

#include <unordered\_set>

using namespace std;

class Credit {

private:

string name;

int sum;

string currency;

int annual\_rate;

public:

Credit();

Credit(string name, int sum, string currency, int annual\_rate);

Credit(const Credit& Credit\_copied);

bool operator<(const Credit& Credit2) const {

return annual\_rate < Credit2.annual\_rate;

};

bool operator==(const Credit& Credit2) const {

return name == Credit2.name && sum == Credit2.sum && currency == Credit2.currency && annual\_rate == Credit2.annual\_rate;

};

friend ostream& operator<<(ostream& os, const Credit& Credit);

friend istream& operator>>(istream& is, Credit& Credit);

friend struct Credit\_Hasher;

};

Credit::Credit() {

this->name = "";

this->sum = 0;

this->currency = "";

this->annual\_rate = 0;

};

Credit::Credit(string name, int sum, string currency, int annual\_rate) {

this->name = name;

this->sum = sum;

this->currency = currency;

this->annual\_rate = annual\_rate;

};

Credit::Credit(const Credit& Credit\_copied) {

this->name = Credit\_copied.name;

this->sum = Credit\_copied.sum;

this->currency = Credit\_copied.currency;

this->annual\_rate = Credit\_copied.annual\_rate;

};

ostream& operator<<(ostream& os, const Credit& Credit) {

fstream file("output.txt", fstream::app);

if (file.is\_open()) {

file << Credit.name << ' ' << Credit.sum << ' ' << Credit.currency << ' ' << Credit.annual\_rate << endl;

}

else {

cout << "Output file didn't open." << endl;

}

os << Credit.name << ' ' << Credit.sum << ' ' << Credit.currency << ' ' << Credit.annual\_rate << '%' << endl;

return os;

};

istream& operator>>(istream& is, Credit& Credit) {

is >> Credit.name;

is >> Credit.sum;

is >> Credit.currency;

is >> Credit.annual\_rate;

return is;

};

struct Credit\_Hasher {

hash<string> shash1;

hash<int> ihash1;

hash<string> shash2;

hash<int> ihash2;

size\_t operator()(const Credit& ob) const

{

const size\_t coef = 2'946'901;

return (

coef \* coef \* coef \* shash1(ob.name) +

coef \* coef \* ihash1(ob.sum) +

coef \* shash2(ob.currency) +

ihash2(ob.annual\_rate)

);

}

};

int main() {

ofstream file\_output("output.txt");

file\_output.close();

ifstream file\_input("input.txt");

vector<Credit> Credits\_vector;

string line;

while (getline(file\_input, line)) {

istringstream ss(line);

Credit to\_vector;

ss >> to\_vector;

Credits\_vector.push\_back(to\_vector);

}

cout << "Vector before sorting: " << endl;

for (int i = 0; i < Credits\_vector.size(); ++i) {

cout << Credits\_vector[i];

}cout << endl;

cout << "Set copied from vector: " << endl;

set<Credit> Credits\_set;

for (int i = 0; i < Credits\_vector.size(); ++i) {

Credits\_set.insert(Credits\_vector[i]);

}

for (const auto &i: Credits\_set) {

cout << i;

}cout << endl;

cout << "Unordered\_set copied from vector: " << endl;

unordered\_set<Credit, Credit\_Hasher> Credits\_unordered\_set;

for (int i = 0; i < Credits\_vector.size(); ++i) {

Credits\_unordered\_set.insert(Credits\_vector[i]);

}

for (const auto &i: Credits\_unordered\_set) {

cout << i;

}cout << endl;

}

**Результат работы программы:**

Vector before sorting:

Oleg 5000 ruble 5%

Egor 3000 dollar 3%

Vlad 4000 euro 4%

Uzbek 2000 ruble 2%

Set copied from vector:

Uzbek 2000 ruble 2%

Egor 3000 dollar 3%

Vlad 4000 euro 4%

Oleg 5000 ruble 5%

Unordered\_set copied from vector:

Vlad 4000 euro 4%

Oleg 5000 ruble 5%

Egor 3000 dollar 3%

Uzbek 2000 ruble 2%

**Вывод:**

В ходе выполнения лабораторной работы были изучены принципы использования объектов своих классов в последовательных контейнерах библиотеки STL.