Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3**

**Тема: Перегрузка операций.**

**Вариант 5**

Дисциплина: «Основы алгоритмизации и программирования»

Семестр 2

Выполнил работу

Студент группы РИС-22-1Б

Батин Владислав Владимирович

Проверил

Доцент кафедры ИТАС

Полякова Ольга Андреевна

2023 год

**Постановка задачи**

Создать класс Time для работы с временными интервалами. Интервал должен быть представлен в виде двух полей: min и sec – типа int. При выводе минуты от секунд отделяются двоеточием. Реализовать:

- вычитание секунд

- сравнение временных интервалов (== и !=)

**Код**

**KL3.25.cpp**

#include "Time.h"

#include <iostream>

using namespace std;

// операция сравнения

void sravn(Time& a, Time& b)

{

int tempa, tempb, am, as, bm, bs;

am = a.get\_min();

as = a.get\_sec();

bm = b.get\_min();

bs = b.get\_sec();

tempa = am \* 60 + as;

tempb = bm \* 60 + bs;

if (tempa == tempb)cout << "\na = b\n";

if (tempa != tempb)

{

if (tempa > tempb)cout << "\na > b\n";

else cout << "\na < b\n";

}

}

void main()

{

Time a; // констурктор без параметров

Time b; // констурктор без параметров

Time c; // констурктор без параметров

cin >> a; // ввод

cin >> b; // ввод

c = a - b;

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

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

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

sravn(a, b);

}

**Time.cpp**

#include "Time.h"

#include <iostream>

using namespace std;

// перегрузка операций присваивания

Time& Time::operator=(const Time&t)

{

// проверка на самоприсваивание

if (&t == this) { return \*this; }

min = t.min;

sec = t.sec;

return \*this;

}

// перегрузка бинарной функции сложения

Time Time::operator-(const Time&t)

{

int temp1 = min \* 60 + sec;

int temp2 = t.min \* 60 + t.sec;

Time p;

p.min = (temp1 - temp2) / 60;

p.sec = (temp1 - temp2) & 60;

return p;

}

// перегрузка глобальной функции ввода

istream& operator>>(istream&in, Time& t)

{

cout << "Vvod min >> "; in >> t.min;

cout << "Vvod sec >> "; in >> t.sec;

return in;

}

// перегрузка глобальной функции вывода

ostream& operator<<(ostream& out, Time& t)

{

return (out << t.min << " : " << t.sec);

}

**Time.h**

#include <iostream>

using namespace std;

class Time

{

int min, sec;

public:

Time() { min = 0; sec = 0; }

Time(int m, int s) { min = m; sec = s; }

Time(const Time& t) { min = t.min; sec = t.sec; }

~Time() {};

int get\_min() { return min; }

int get\_sec() { return sec; }

void set\_min(int m) { min = m; }

void set\_sec(int s) { sec = s; }

// перегруженные операции

Time& operator=(const Time&);

Time operator-(const Time&);

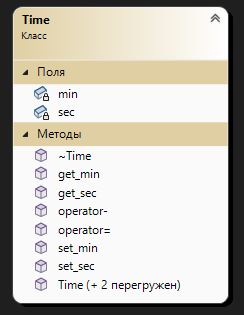
// глобальные функции вводв а и вывода

friend istream& operator >>(istream& in, Time& t);

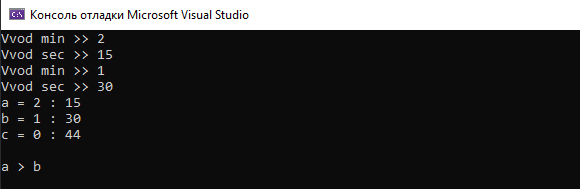
friend ostream& operator <<(ostream& out, Time& t);

};

**Диаграмма класса**

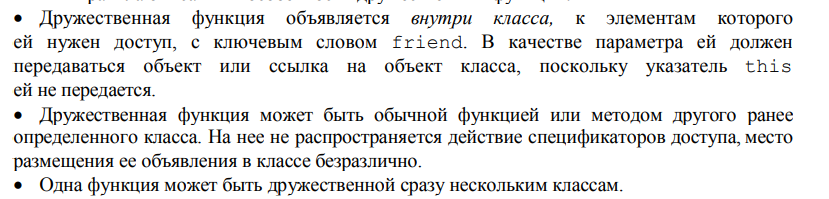


**Результат выполнения программы**

****

**Ответы на вопросы**

**1**. если сумма у одного объекта совпадет с суммой другого, то будет получено true, иначе – false

**2**. 

**3**. Как метод, как внешнюю функцию.

**4**. 0.

**5**. 1.

**6**. 1. Ее операнд – вызвавший ее же объект.

**7**. 2 операнда типа класса.

**8**. префиксная приоритетнее, в нее должно передаваться значение.

**9**. Class&operator(const Class&other).

**10**. Объект.

**11**. При помощи дружественной глобальной ф-ии.

**12**. Префиксный инкремент.

**13**. Перегруженная дружественная ф-ия.

**14**. Если а действительно меньше b вернется 1.

**15**. Если а действительно больше b вернется 1.