Пермский Национальный Исследовательский  
Политехнический Университет

**Лабораторная работа № 3**

Информатика

за 2 семестр

Вариант № 1

Выполнил:

Студент группы РИС 20-1-бз

Акиев В.В.

20-ЭТФ-641

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

2021

**Перегрузка операций.**

1. **Цель задания**.

1.1 Создание консольного приложения, состоящего из нескольких файлов в системе программирования VisualStudio.

1.2. Использование перегруженных операций в классах.

1. **Задание.**

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

- сложение временных интервалов (учесть, что в минуте не может быть более 60 секунд)

- сравнение временных интервалов (==)

1. **Файл Лабораторная\_3.cpp**

#include <iostream>

#include "time\_class.h"

using namespace std;

int main()

{

Time time1, time2;

cout << "Enter time objects.\nTime #1:\n";

cin >> time1;

cout << "Time #2:\n";

cin >> time2;

cout << "You enter 2 time objects: " << time1 << " " << time2 << "\n"

<< "time#1 + time#2 = " << time1 << " + " << time2 << " = " << time1 + time2 << "\n"

<< "Equality: " << time1 << (time1 == time2 ? "" : " NOT ") << "equals " << time2 << "\n";

cin.ignore();

cin.get();

return 0;

}

**Файл time\_class.cpp**

#include <iostream>

#include <iomanip>

#include <fstream>

#include "time\_class.h"

Time::Time() : minutes(0), seconds(0) {}

Time::Time(int m, int s) : minutes(m + s / 60), seconds(s % 60) {}

Time::Time(const Time& t) : minutes(t.minutes), seconds(t.seconds) {}

Time::~Time() {}

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

{

minutes = t.minutes;

seconds = t.seconds;

return \*this;

}

std::ostream& operator<<(std::ostream& out, const Time& t)

{

return out << std::setfill('0') << std::setw(2) << t.minutes << ":"

<< std::setw(2) << t.seconds << std::setfill(' ');

}

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

{

int m, s;

std::cout << "\tMinutes: "; in >> m;

std::cout << "\tSeconds: "; in >> s;

t = Time(m, s);

return in;

}

bool operator==(const Time& time1, const Time& time2)

{

return (time1.minutes == time2.minutes) && (time1.seconds == time2.seconds);

}

Time operator+(const Time& time1, const Time& time2)

{

return Time(time1.minutes + time2.minutes, time1.seconds + time2.seconds);

}

**Файл time\_class.h**

#pragma once

#ifndef TIME\_CLASS\_H

#define TIME\_CLASS\_H

#include <iosfwd>

class Time

{

friend std::ostream& operator<<(std::ostream&, const Time&);

friend std::istream& operator>>(std::istream&, Time&);

friend bool operator==(const Time&, const Time&);

friend Time operator+(const Time&, const Time&);

public:

Time();

Time(int, int);

Time(const Time&);

Time operator=(const Time&);

~Time();

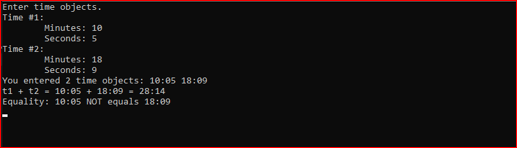
private:

int minutes, seconds;

};

#endif

1. **Результат выполнения программы**



1. **Контрольные вопросы.**
2. **Для чего используются дружественные функции и классы?**

Дружественные функции применяются для доступа к скрытым полям класса и представляют собой альтернативу методам.

1. **Сформулировать правила описания и особенности дружественных функций.**

Дружественные функции - это функции, которые не являются членами класса, однако имеют доступ к его закрытым членам - переменным и функциям, которые имеют спецификатор private.

1. **Каким образом можно перегрузить унарные операции?**
   * + Как компонентную функцию класса
     + Как внешнюю (глобальную) функцию
2. **Сколько операндов должна иметь унарная функция-операция, определяемая внутри класса?**

Унарная функция-операция, определяемая внутри класса, должна быть представлена с помощью нестатического метода без параметров, при этом операндом является вызвавший ее объект

1. **Сколько операндов должна иметь унарная функция-операция, определяемая вне класса?**

Если функция определяется вне класса, она должна иметь один параметр типа класса

1. **Сколько операндов должна иметь бинарная функция-операция, определяемая внутри класса?**

Бинарная функция-операция, определяемая внутри класса, должна быть представлена с помощью нестатического метода с параметрами, при этом вызвавший ее объект считается первым операндом

1. **Сколько операндов должна иметь бинарная функция-операция, определяемая вне класса?**

Если функция определяется вне класса, она должна иметь два параметра типа класса

1. **Чем отличается перегрузка префиксных и постфиксных унарных операций?**

Префиксные операторы ++x (--x) записывают в переменную x изменённое значение и его же возвращают вместо себя (в исходное выражение). Постфиксные операторы x++ (x--) возвращают вместо себя текущее значение переменной х, а потом записывают в x новое значение.

1. **Каким образом можно перегрузить операцию присваивания?**

Операция присваивания определена в любом классе по умолчанию как поэлементное копирование. Эта операция вызывается каждый раз, когда одному существующему объекту присваивается значение другого.

1. **Что должна возвращать операция присваивания?**

операция-функция должна возвращать ссылку на объект, для которого она вызвана, и принимать в качестве параметра единственный аргумент — ссылку на присваиваемый объект

1. **Каким образом можно перегрузить операции ввода-вывода?**

Операции ввода-вывода operator>> и operator<< всегда реализуются как внешние дружественные функции, т. к. левым операндом этих операций являются потоки.