Московский Авиационный Институт

(Национальный исследовательский Университет)

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

**Лабораторная работа**

**по курсу «ООП»**

**Тема:**

**Простые классы.**

|  |  |
| --- | --- |
| Студент: | Николаев В.А. |
| Группа: | М80-206Б-18 |
| Преподаватель: | Журавлев А.А. |
| Вариант: | 14 |
| Оценка: |  |
| Дата: |  |

Москва

2019

**1. Код программы на языке C++:**

**timepoint.h:**

#include <iostream>

class TimePoint

{

public:

long long hours;

long long min;

long long sec;

bool compare(TimePoint moment, double\* answ);

void secadding(long long addsec);

void secdiff(long long diffsec);

void minadding(long long addmin);

long long trans\_to\_sec();

long long trans\_to\_min();

};

TimePoint operator + (TimePoint first, TimePoint second);

TimePoint operator - (TimePoint first, TimePoint second);

std::istream& operator>> (std::istream &npt, TimePoint &point);

std::ostream& operator<< (std::ostream &out, TimePoint &point);

bool operator > (TimePoint first, TimePoint second);

bool operator < (TimePoint first, TimePoint second);

bool operator == (TimePoint first, TimePoint second);

unsigned long long operator "" \_mult(unsigned long long npt);

**timepoint.cpp:**

#include <iostream>

#include <cmath>

#include "timepoint.h"

void TimePoint::secadding(long long addsec)

{

this->sec += addsec % 60;

this->min += (addsec / 60) % 60;

this->hours += addsec / 3600;

if (this->sec >= 60) {

this->sec = this->sec % 60;

this->min += 1;

}

if (this->min >= 60) {

this->min = this->min % 60;

this->hours += 1;

}

this->hours = this->hours % 24;

}

void TimePoint::secdiff(long long diffsec)

{

this->sec -= diffsec;

if (this->sec < 0) {

this->min -= 1;

this->min -= diffsec / 60;

this->sec = this->sec % 60 + 60;

}

if (this->min < 0) {

this->hours -= diffsec / 3600;

this->min = this->min % 60 + 60;

}

this->sec = this->sec % 60;

this->min = this->min % 60;

this->hours = abs(this->hours % 24);

}

void TimePoint::minadding(long long addmin)

{

this->min += addmin % 60;

this->hours += addmin / 60;

if (this->min >= 60) {

this->min = this->min % 60;

this->hours += 1;

}

this->hours = this->hours % 24;

}

bool TimePoint::compare(TimePoint moment, double\* answ) {

long long a = this->trans\_to\_sec();

long long b = moment.trans\_to\_sec();

\*answ = double(a) / double(b);

return a > b;

}

long long TimePoint::trans\_to\_sec() {

this->min = (this->hours \* 60) + this->min;

this->sec = (this->min \* 60) + this->sec;

return this->sec;

}

long long TimePoint::trans\_to\_min() {

this->min = (this->hours \* 60) + this->min;

if (this->sec >= 30) {

this->min += 1;

}

return this -> min;

}

TimePoint operator + (TimePoint first, TimePoint second) {

first.sec += second.sec;

first.min += second.min + (first.sec / 60);

first.hours += second.hours + (first.min / 60);

first.min = first.min % 60;

first.sec = first.sec % 60;

first.hours = first.hours % 24;

return first;

}

TimePoint operator - (TimePoint first, TimePoint second)

{

first.sec -= second.sec;

if (first.sec < 0) {

first.min -= 1;

first.sec += 60;

}

first.min -= second.min;

if (first.min < 0) {

first.hours -= 1;

first.min += 60;

}

first.hours -= second.hours;

first.hours = abs(first.hours);

return first;

}

bool operator > (TimePoint first, TimePoint second)

{

return first.sec + first.min \* 60 + first.hours \* 3600 > second.sec + second.min \* 60 + second.hours \* 3600;

}

bool operator < (TimePoint first, TimePoint second)

{

return first.sec + first.min \* 60 + first.hours \* 3600 < second.sec + second.min \* 60 + second.hours \* 3600;;

}

bool operator == (TimePoint first, TimePoint second)

{

return first.sec + first.min \* 60 + first.hours \* 3600 == second.sec + second.min \* 60 + second.hours \* 3600;

}

unsigned long long operator "" \_mult(unsigned long long npt)

{

return npt\*60;

}

std::istream& operator>> (std::istream &npt, TimePoint &point)

{

char tmp;

npt >> point.hours;

npt >> tmp;

npt >> point.min;

npt >> tmp;

npt >> point.sec;

return npt;

}

std::ostream& operator<< (std::ostream &out, TimePoint &point)

{

out << point.hours << " часов " << point.min << " минут " << point.sec << " секунд ";

return out;

}

**main.cpp:**

#include <iostream>

#include "timepoint.h"

int main()

{

std::cout << "Меню\n1)Вычисление разницы между двумя моментами времени\n2)Сумма моментов времени\n3)Сложение момента времени и заданного количества секунд\n4)Вычитание из момента времени заданного количества секунд\n5)Вычисление, во сколько раз один момент времени больше(меньше) другого\n6)Сравнение моментов времени\n7)Перевод в секунды\n8)Перевод в минуты\n9)Перевод из секунд в формат час:минута:секунда\n10)Перевод из минут в формат час:минута:секунда\n11)Пользовательский литерал\n";

TimePoint res;

std::string npt;

std::cin >> npt;

if (npt == "1") {

std::cout << "Ввод осуществляется в формате час:минута:секунда\nВведите момент, из которого необходимо произвести вычитание: ";

TimePoint first;

TimePoint second;

std::cin >> first;

std::cout << "Введите момент времени, который необходимо вычесть: ";

std::cin >> second;

first = first - second;

std::cout << first;

}

if (npt == "2") {

std::cout << "Ввод осуществляется в формате час:минута:секунда\nВведите первый момент времени: ";

TimePoint first;

std::cin >> first;

std::cout << "Введите второй момент времени: ";

TimePoint second;

std::cin >> second;

first = first + second;

std::cout << first;

}

if (npt == "3") {

std::cout << "Ввод осуществляется в формате час:минута:секунда\nВведите момент времени: ";

TimePoint moment;

long long addsec;

std::cin >> moment;

std::cout << "Введите количество секунд: ";

std::cin >> addsec;

moment.secadding(addsec);

std::cout << moment;

}

if (npt == "4") {

std::cout << "Ввод осуществляется в формате час:минута:секунда\nВведите момент времени: ";

TimePoint moment;

std::cin >> moment;

long long diffsec;

std::cout << "Введите количество секунд: ";

std::cin >> diffsec;

moment.secdiff(diffsec);

std::cout << moment;

}

if (npt == "5") {

std::cout << "Ввод осуществляется в формате час:минута:секунда\nВведите первый момент времени: ";

TimePoint first;

double answ = 0;

std::cin >> first;

std::cout << "Введите второй момент времени: ";

TimePoint second;

std::cin >> second;

if (first.compare(second, &answ) && answ != 1) {

std::cout << "Первый момент больше второго в " << answ << " раз\n";

}

else if (answ == 1) {

std::cout << "Они равны\n";

}

else {

std::cout << "Первый момент меньше второго в " << answ << " раз\n";

}

}

if (npt == "6") {

std::cout << "Ввод осуществляется в формате час:минута:секунда\nВведите первый момент времени: ";

TimePoint first;

std::cin >> first;

std::cout << "Введите второй момент времени: ";

TimePoint second;

std::cin >> second;

long long a = 0;

long long b = 0;

if (first > second) {

std::cout << "Первый момент времени больше второго";

}

if (first < second) {

std::cout << "Второй момент времени больше первого";

}

if (first == second) {

std::cout << "Они равны";

}

}

if (npt == "7") {

std::cout << "Ввод осуществляется в формате час:минута:секунда\nВведите момент времени: ";

TimePoint moment;

std::cin >> moment;

long long answ = moment.trans\_to\_sec();

std::cout << "Результат:\n" << answ << " секунд ";

}

if (npt == "8") {

std::cout << "Ввод осуществляется в формате час:минута:секунда\nВведите момент времени: ";

TimePoint moment;

std::cin >> moment;

long long answ = moment.trans\_to\_min();

std::cout << "Результат:\n" << answ << " минут ";

}

if (npt == "9") {

long long addsec;

std::cout << "Введите количество секунд: ";

std::cin >> addsec;

res.secadding(addsec);

std::cout << res;

}

if (npt == "10") {

long long addmin;

std::cout << "Введите количество минут: ";

std::cin >> addmin;

res.minadding(addmin);

std::cout << res;

}

if (npt == "11") {

std::cout << 42\_mult;

}

}

**CmakeLists.txt:**

project(lab2)

add\_executable(lab2

main.cpp

timepoint.cpp

timepoint.h

)

**2. Ссылка на репозиторий на GitHub.**

**https://github.com/a1dv/oop\_exercise\_02.git**

**3. Набор тестов.**

test\_00.test:

22:13:52

14:37:58

8833123

test\_01.test:

14:59:22

18:34:21

12312

test\_02.test:

1:1:1

23:59:59

14124124

**4. Результаты выполнения тестов.**

test\_00.result:

Сложение:

Результат:

12 часов 51 минут 50 секунд

Вычитание:

Результат:

7 часов 35 минут 54 секунд

Добавление секунд:

Результат:

3 часов 52 минут 35 секунд

Вычитание секунд:

Результат:

8 часов 35 минут 9 секунд

Сравнение:

Первый момент больше второго в 1.51927 раз

Перевод в секунды:

Результат:

80032 секунд

Перевод в минуты:

Результат:

1334 минут

Перевод из секунд:

Результат:

5 часов 38 минут 43 секунд

Перевод из минут:

Результат:

2 часов 43 минут 0 секунд

test\_01.result:

Сложение:

Результат:

9 часов 33 минут 43 секунд

Вычитание:

Результат:

20 часов 25 минут 1 секунд

Добавление секунд:

Результат:

18 часов 24 минут 34 секунд

Вычитание секунд:

Результат:

11 часов 34 минут 10 секунд

Сравнение:

Первый момент меньше второго в 1.23904 раз

Перевод в секунды:

Результат:

53962 секунд

Перевод в минуты:

Результат:

899 минут

Перевод из секунд:

Результат:

3 часов 25 минут 12 секунд

Перевод из минут:

Результат:

13 часов 12 минут 0 секунд

test\_02.result:

Сложение:

Результат:

1 часов 1 минут 0 секунд

Вычитание:

Результат:

1 часов 1 минут 2 секунд

Добавление секунд:

Результат:

12 часов 23 минут 5 секунд

Вычитание секунд:

Результат:

11 часов 38 минут 57 секунд

Сравнение:

Первый момент меньше второго в 23.5998 раз

Перевод в секунды:

Результат:

3661 секунд

Перевод в минуты:

Результат:

61 минут

Перевод из секунд:

Результат:

11 часов 22 минут 4 секунд

Перевод из минут:

Результат:

10 часов 4 минут 0 секунд

**5. Объяснение результатов работы программы.**

1) Ввод осуществляется через поток стандартного ввода

2) Вывод осуществляется через поток стандартного вывода.

3) Ввод осуществляется в формате час:минута:секунда.

5) С помощью метода summaring класса Timepoint производится сложение двух моментов времени

6) С помощью метода difference класса Timepoint вычисляется разность между двумя моментами времени

7) С помощью метода secadding класса Timepoint к заданному моменту времени прибавляется заданное количество секунд

8) С помощью метода difference класса Timepoint из заданного момента времени вычитается заданное количество секундах

9) С помощью метода trans\_to\_sec класса Timepoint производится сравнение двух моментов времени

10) С помощью метода trans\_to\_sec класса Timepoint производится перевод в секунды

11) С помощью метода trans\_to\_min класса Timepoint производится перевод в минуты

**6. Вывод.**

Изучил перегрузку операторов и пользовательский литерал.