Пермский Национальный Исследовательский  
Политехнический Университет

**Лабораторная работа №10**

Основы алгоритмизации и программирования

за 2 семестр

Вариант № 1

Выполнил:

Студент группы РИС 20-1-бз

Курганов Н.В.

20-ЭТФ-631

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

2022

Сохранение данных в файле с использованием потоков

1. **Цель задания**.

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

1.2. Разработка программы, в которой данные сохраняются в файле, корректируются и выводятся из файла на печать. Работа с файлом осуществляется с использованием потоковых классов.

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. Удалить все записи равные заданному значению.

2. Увеличить все записи с заданным значением на 1 минуту 30 секунд.

3. Добавить К записей после элемента с заданным номером.

1. **Создаем:**

**Файл Lab10\_main.cpp**

#include <iostream>

#include <string>

#include "time\_class.h"

#include "time\_funcs.h"

int main()

{

std::cout << "Welcome to Lab 10!\nEnter filename for work: ";

std::string filename;

std::cin >> filename;

int x = 0;

do {

std::cout << "\n\nMenu:\n"

<< "1. Add to file\n"

<< "2. Print file\n"

<< "3. Delete values\n"

<< "4. Increment +1:30\n"

<< "5. Insert values\n"

<< "0. Exit\n"

<< ">> ";

std::cin >> x;

switch (x) {

case 0: break;

case 1: add\_to\_end(filename); break;

case 2: print(filename); break;

case 3: delete\_values(filename); break;

case 4: increment\_values(filename); break;

case 5: insert\_after(filename); break;

default: std::cout << "Wrong input!\n";

}

} while (x != 0);

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::operator=(const Time& t)

{

minutes = t.minutes;

seconds = t.seconds;

return \*this;

}

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

{

\*this = Time(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;

}

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

{

out << t.minutes << " " << t.seconds << "\n";

return out;

}

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

{

in >> t.minutes >> t.seconds;

return in;

}

bool operator==(const Time& t1, const Time& t2)

{

return (t1.minutes == t2.minutes) && (t1.seconds == t2.seconds);

}

bool operator!=(const Time& t1, const Time& t2)

{

return !(t1 == t2);

}

bool operator<(const Time& t1, const Time& t2)

{

return (t1.minutes < t2.minutes) ||

((t1.minutes == t2.minutes) && (t1.seconds < t2.seconds));

}

bool operator>(const Time& t1, const Time& t2)

{

return !(t1 < t2) && (t1 != t2);

}

Time operator+(const Time& t1, const Time& t2)

{

return Time(t1.minutes + t2.minutes, t1.seconds + t2.seconds);

}

Time operator/(const Time& t, int x)

{

int secs = (t.minutes \* 60 + t.seconds) / x;

return Time(0, secs);

}

**Файл Time\_class.h**

#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 std::ofstream& operator<<(std::ofstream&, const Time&);

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

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

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

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

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

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

friend Time operator/(const Time&, int);

public:

Time();

Time(int, int);

Time(const Time&);

Time operator=(const Time&);

Time& operator+=(const Time&);

private:

int minutes, seconds;

};

#endif

**Файл Time\_funcs.cpp**

#include <iostream>

#include <fstream>

#include <string>

#include <stdio.h>

#include "time\_class.h"

#include "time\_funcs.h"

void add\_to\_end(const std::string & filename)

{

std::ofstream out(filename, std::ios::out | std::ios::app);

if (!out.is\_open()) {

std::cout << "Can't open file " << filename << "!\n";

return;

}

std::cout << "Enter number of time objects: ";

int n;

std::cin >> n;

for (int i = 0; i < n; ++i) {

std::cout << "Time #" << i + 1 << ":\n";

Time t;

std::cin >> t;

out << t;

}

}

void print(const std::string & filename)

{

std::ifstream in(filename);

if (!in.is\_open()) {

std::cout << "Can't open file " << filename << "!\n";

return;

}

Time t;

std::cout << "File " << filename << " contains time objects:\n";

while (in >> t) {

std::cout << t << "\n";

}

}

void delete\_values(const std::string & filename)

{

std::ifstream in(filename);

if (!in.is\_open()) {

std::cout << "Can't open file " << filename << "!\n";

return;

}

std::string tempfile = "temp.txt";

std::ofstream temp(tempfile);

std::cout << "Enter time value for delete:\n";

Time del\_t;

std::cin >> del\_t;

Time t;

while (in >> t) {

if (t != del\_t)

temp << t;

}

in.close();

temp.close();

std::remove(filename.c\_str());

std::rename(tempfile.c\_str(), filename.c\_str());

}

void increment\_values(const std::string & filename)

{

std::ifstream in(filename);

if (!in.is\_open()) {

std::cout << "Can't open file " << filename << "!\n";

return;

}

std::string tempfile = "temp.txt";

std::ofstream temp(tempfile);

std::cout << "Incrementing all values:\n";

Time t;

while (in >> t) {

std::cout << t << " + " << Time(1, 30) << " = " << t + Time(1, 30) << "\n";

temp << t + Time(1, 30);

}

in.close();

temp.close();

std::remove(filename.c\_str());

std::rename(tempfile.c\_str(), filename.c\_str());

}

void insert\_after(const std::string & filename)

{

std::ifstream in(filename);

if (!in.is\_open()) {

std::cout << "Can't open file " << filename << "!\n";

return;

}

std::string tempfile = "temp.txt";

std::ofstream temp(tempfile);

std::cout << "Enter number of new time objects: ";

int k;

std::cin >> k;

std::cout << "Enter index for insert: ";

int n;

std::cin >> n;

Time t;

while ((n > 0) && (in >> t)) {

temp << t;

--n;

}

for (int i = 0; i < k; ++i) {

std::cin >> t;

temp << t;

}

while (in >> t)

temp << t;

in.close();

temp.close();

std::remove(filename.c\_str());

std::rename(tempfile.c\_str(), filename.c\_str());

}

**Файл Time\_funcs.h**

#ifndef TIME\_FUNCS\_H

#define TIME\_FUNCS\_H

#include <string>

void add\_to\_end(const std::string & filename);

void print(const std::string & filename);

void delete\_values(const std::string & filename);

void increment\_values(const std::string & filename);

void insert\_after(const std::string & filename);

#endif

**Результат работы программы**

