МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСТИТЕТ)

ЛАБОРАТОРНАЯ РАБОТА №1 по курсу

объектно-ориентированное программирование I семестр, 2021/22 уч. год

Студент *Морозов Артем Борисович, группа М80-208Б-20*

Преподаватель *Дорохов Евгений Павлович*

Цель:

- Изучение системы сборки на языке C++, изучение систем контроля версии.
- Изучение основ работы с классами в C++;

Порядок выполнения работы

1. Ознакомиться с теоретическим материалом.

2. Получить у преподавателя вариант задания.

3. Реализовать задание своего варианта в соответствии с поставленными требованиями.

4. Подготовить тестовые наборы данных.

5. Создать репозиторий на GitHub.

6. Отправить файлы лабораторной работы в репозиторий.

7. Отчитаться по выполненной работе путём демонстрации работающей программы на тестовых наборах данных (как подготовленных самостоятельно, так и предложенных преподавателем) и ответов на вопросы преподавателя (как из числа контрольных, так и по реализации программы).

Требования к программе

Разработать программу на языке C++ согласно варианту задания. Программа на C++ должна собираться с помощью системы сборки CMake. Программа должна получать данные из стандартного ввода и выводить данные в стандартный вывод.

Необходимо настроить сборку лабораторной работы с помощью CMake. Собранная программа должна называться **oop_exercise_01** (в случае использования Windows **oop_exercise_01.exe)**

Необходимо зарегистрироваться на GitHub (если студент уже имеет регистрацию на GitHubто можно использовтаь ее) и создать репозитарий для задания лабораторной работы.

Преподавателю необходимо предъявить ссылку на публичный репозиторий на Github. Имя репозитория должно быть https://github.com/*login*/oop_exercise_01

Где login – логин, выбранный студентом для своего репозитория на Github.

Репозиторий должен содержать файлы:

- main.cpp //файл с заданием работы
- CMakeLists.txt // файл с конфигураций CMake
- test_xx.txt // файл с тестовыми данными. Где xx – номер тестового

набора 01, 02 , … Тестовых наборов должно быть больше 1.

- report.doc // отчет о лабораторной работе

Описание программы

Исходный код лежит в 3 файлах:

1. main.cpp - исполняемый код.
2. TimePoint.h - специальный файл .h, содержащий прототипы используемых мною функций.
3. TimePoint.cpp - реализация функций для моего задания.
4. CMakeLists.txt - специальный дополнительный файл типа CMakeLists.

Дневник отладки

Во время выполнения лабораторной работы программа не нуждалась в отладке, все ошибки компиляции были исправлены с первой попытки. После их исправления программа работала так, как было задумано изначально.

Недочёты

Недочётов не было обнаружено.

Выводы

Данная лабораторная работа помогла мне использовать полученные на лекциях теоретические знания на практике, и я написал простенький полностью работающий класс.

Исходный код

TimePoint.h

```
#ifndef TIMEPOINT_H
#define TIMEPOINT_H
#include <iostream>
class TimePoint {
    public:
        TimePoint();
        TimePoint(int h, int m, int s);
        TimePoint(std::istream &is);
        TimePoint(const TimePoint &other);
        void Difference(const TimePoint &other);
        void Sum(const TimePoint& other);
        void AddSeconds(int s);
        void RemoveSeconds(int s);
        int IsBigger(const TimePoint &other);
        void Compare(const TimePoint &other);
        int ToSeconds();
        int ToMinutes();
        void Print(std::ostream &os);
        ~TimePoint();
    private:
        int hours;
        int minutes;
        int seconds;
};
#endif
```

TimePoint.cpp

```
#include "TimePoint.h"

TimePoint::TimePoint() {
    hours = 0;
    minutes = 0;
    seconds = 0;
    std:: cout << "The default time-ojbect has been created" << std:: endl;
}

TimePoint::TimePoint(int h, int m, int s) {
    if (h >= 0 && m >= 0 && s >= 0) {
        hours = h;
        minutes = m;
        seconds = s;
    }
    else {
        std:: cout << "Please enter positive numbers!" << std:: endl;
    }
    std:: cout << "The time-object according to your parameters has been created" << std:: endl;
}

TimePoint::TimePoint(std::istream &is) {
    std:: cout << "Please enter your time-object data: " << std:: endl;
    is >> hours >> minutes >> seconds;
    if ((hours < 0 || hours > 23) || (minutes < 0 || minutes > 59) || (seconds < 0 || seconds > 59)) {
        std:: cout << "Invalind input. Enter again!" << std:: endl;
        is >> hours >> minutes >> seconds;
    }
    std:: cout << "The time-object has been created via istream" << std:: endl;
}

TimePoint::TimePoint(const TimePoint& other) {
    hours = other.hours;
    minutes = other.minutes;
    seconds = other.seconds;
    std:: cout << "The copy of your time-object has been created" << std:: endl;
}

void TimePoint::Difference(const TimePoint &other) {
    int x = hours * 3600 + minutes * 60 + seconds;
    int y = other.hours * 3600 + other.minutes * 60 + other.seconds;
    int dhours, dminutes, dseconds;
    if ((hours > other.hours) || (hours == other.hours && minutes > other.minutes) || (hours ==
other.hours && minutes == other.minutes && seconds > other.seconds)) {
        int z = x - y;
        dhours = z / 3600;
```

```
        dminutes = (z % 3600) / 60;
        dseconds = (z % 3600) - (dminutes * 60);
    }
    else {
        int z = y - x;
        dhours = z / 3600;
        dminutes = (z % 3600) / 60;
        dseconds = (z % 3600) - (dminutes * 60);
    }
    std:: cout << "The difference between your time-objects is: " << dhours << ":" << dminutes << ":"
<< dseconds << std:: endl;
}

void TimePoint::Sum(const TimePoint& other) {
    int x = hours * 3600 + minutes * 60 + seconds;
    int y = other.hours * 3600 + other.minutes * 60 + other.seconds;
    int z = x + y;
    int dhours = z / 3600;
    int dminutes = (z % 3600) / 60;
    int dseconds = (z % 3600) - (dminutes * 60);
    std:: cout << "The sum of your time-objects is: " << dhours << ":" << dminutes << ":" <<
dseconds << std:: endl;
}

void TimePoint::AddSeconds(int s) {
    if (s < 0) {
        std:: cout << "Please enter positive number!" << std:: endl;
    }
    else {
        int x = hours * 3600 + minutes * 60 + seconds + s;
        hours = x / 3600;
        minutes = ((x % 3600) / 60);
        seconds = (x % 3600) - (((x % 3600) / 60) * 60);
        std:: cout << "After adding seconds your time is: "  << hours << ":" << minutes << ":" <<
seconds << std:: endl;
    }
}

void TimePoint::RemoveSeconds(int s) {
    if (s < 0) {
        std:: cout << "Please enter positive number!" << std:: endl;
    }
    else {
        int x = hours * 3600 + minutes * 60 + seconds - s;
        hours = x / 3600;
        minutes = ((x % 3600) / 60);
        seconds = (x % 3600) - (((x % 3600) / 60) * 60);
        std:: cout << "After removing seconds your time is: "  << hours << ":" << minutes << ":" <<
seconds << std:: endl;
    }
}
```

```
int TimePoint::IsBigger(const TimePoint &other ) {
    int x = hours * 3600 + minutes * 60 + seconds;
    int y = other.hours * 3600 + other.minutes * 60 + other.seconds;
    if ((hours > other.hours) || (hours == other.hours && minutes > other.minutes) || (hours ==
other.hours && minutes == other.minutes && seconds > other.seconds)) {
        return x / y;
    }
    return y / x;
}

void TimePoint::Compare(const TimePoint &other) {
    if ((hours > other.hours) || (hours == other.hours && minutes > other.minutes) || (hours ==
other.hours && minutes == other.minutes && seconds > other.seconds)) {
        std:: cout << "The first time is more that second time!" << std:: endl;
    }
    else if (hours == other.hours && minutes == other.minutes && seconds == other.seconds) {
        std:: cout << "Times are equal!" << std:: endl;
    }
    else {
        std:: cout << "The second time is more that first time!" << std:: endl;
    }
}

int TimePoint::ToSeconds() {
    return hours * 3600 + minutes * 60 + seconds;
}

int TimePoint::ToMinutes() {
    int z = hours * 3600 + minutes * 60 + seconds;
    int m = z / 60;
    if (z % 60 == 0) {
        return m;
    }
    else {
        if (z % 60 >= 30) {
            return m + 1;
        }
    }
    return m;
}

void TimePoint::Print(std::ostream& os) {
    os << "Your current time is: " << hours << ":" << minutes << ":" << seconds << std:: endl;
}

TimePoint::~TimePoint() {
    std:: cout << "FROM DESTRUCTOR: Your time-object has been deleted" << std:: endl;
}
```

main.cpp

```
#include "TimePoint.h"

int main () {
    TimePoint a(std:: cin);
    TimePoint b(12, 38, 40);
    TimePoint c(20, 20, 41);
    TimePoint d(c);
    c.Difference(d);
    b.Sum(c);
    d.AddSeconds(3600);
    c.Print(std:: cout);
    d.Print(std:: cout);
    b.RemoveSeconds(3240);
    b.Print(std:: cout);
    a.Compare(c);
    TimePoint e(06, 00, 00);
    TimePoint f(18, 00, 00);
    std:: cout << "The diffrence between times in their division is: " << e.IsBigger(f)
<< std:: endl;
    std:: cout << "Your time in minutes is: " << a.ToMinutes() << std:: endl;
    std:: cout << "Your time is seconds is: " << a.ToSeconds() << std:: endl;
    return 0;
}
```